ISSN 0542 9951

4
1996

NAJSTARSZE PISMO INFORMATYCZNE W POLSCE – UKAZUJE SIĘ OD 1965 ROKU

P. 1877/96

informatyka

miesięcznik profesjonalistów

Kadra naukowa pilnie poszukiwana

Krystyna Karwicka

Każdego dnia w prasie można znaleźć co najmniej kilkanaście ofert pracy dla informatyków. Poszukiwani są administratorzy systemów i ich projektanci, pracownicy działów sprzedaży i serwisu, programiści pracujący we wszystkich istniejących językach programowania, nawet tych najstarszych. Ogłoszenia te powtarzają się wielokrotnie, co oznacza, że stanowiska nadal nie są obsadzone. Trwa też swoista karuzela kadry. Nawet ci, którzy nie najlepiej bywali oceniani w poprzedniej firmie, odchodząc natychmiast znajdują nową pracę, i to z reguły na lepszych warunkach. Nagminnie występuje podkupywanie pracowników firm komputerowych. Przyczyna jest prosta: rozwój naszego rynku informatycznego sprawia, iż podaż na rynku pracy nie nadąża za popytem. W niektórych specjalizacjach podaż jest nieomal zerowa. Aby wykształcić projektantów systemów, integratorów i analityków systemowych nie starcza wiedzy zdobywanej na uczelniach, trzeba jeszcze stałej współpracy z praktykami, którzy potrafią przekazywać swoją wiedzę. A takich nadal jest u nas jeszcze jak na lekarstwo.

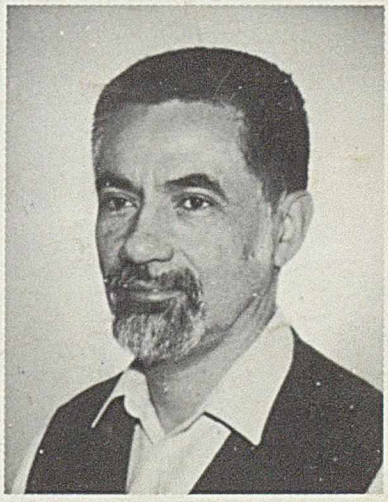
Przyczyny tego, że podaż na rynku pracy nie nadąża za popytem, zostały dokładnie określone na I Kongresie Informatyki Polskiej (Poznań, 1–3 grudnia 1994 r.). Studia na wszystkich kierunkach informatycznych państwowych wyższych uczelni – matematycznym, technicznym i ekonomicznym – kończy rocznie zaledwie kilkuset studentów. Wprawdzie coraz większą rolę w kształceniu informatyków zaczynają odgrywać szkoły niepaństwowe, a także powstające na państwowych wyższych uczelniach studia podyplomowe i zaoczne, ale na ich absolwentów jeszcze trochę trzeba poczekać.

Wszystkie wyższe szkoły, zwłaszcza państwowe, cierpią na niedobór kadry dydaktycznej. W Raporcie kongresowym zostało to określone w sposób jasny: *Najpoważniejszym zagrożeniem dla przyszłości rozwoju informatyzacji w Polsce jest zmniejszenie się liczebności kadr w uczelniach i obniżenie poziomu nauczania oraz brak motywacji finansowych dla nowych kadr*. Wszystkie uczelnie alarmują, że liczebność kadry dydaktycznej w zastraszającym tempie ubywa. Przyczyna jest bardzo prosta: absolwent, a nawet student kierunku informatycznego na dzień dobry otrzymuje w firmie pensję nieraz i trzykrotnie przewyższającą apanaże pracowników naukowych. Wśród osób, które decydują się podjąć pracę dydaktyczną, ze świeczką trzeba szukać takiej, która by nie dorabiała poza uczelnią różnymi sposobami do akademickiej pensyjki. W niektórych specjalnościach najbardziej na rynku pracy deficytowych, takich jak analitycy systemów, integratorzy, administratorzy sieci – uczelnie nie są w stanie znaleźć chętnych do prowadzenia zajęć dydaktycznych. Zresztą i w firmach są oni na wagę złota. Chociaż coraz powszechniej informatyką interesuje się młodzież szkół średnich, a także podstawowych, coraz trudniej jest o nauczycieli informatyki.

dokończenie na s. 34

Od teorii do praktyki

Z prof. dr. hab. Janem Madeyem – prodziekanem Wydziału Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego – na temat zmian w uniwersyteckim nauczaniu informatyki – rozmawia Krystyna Karwicka



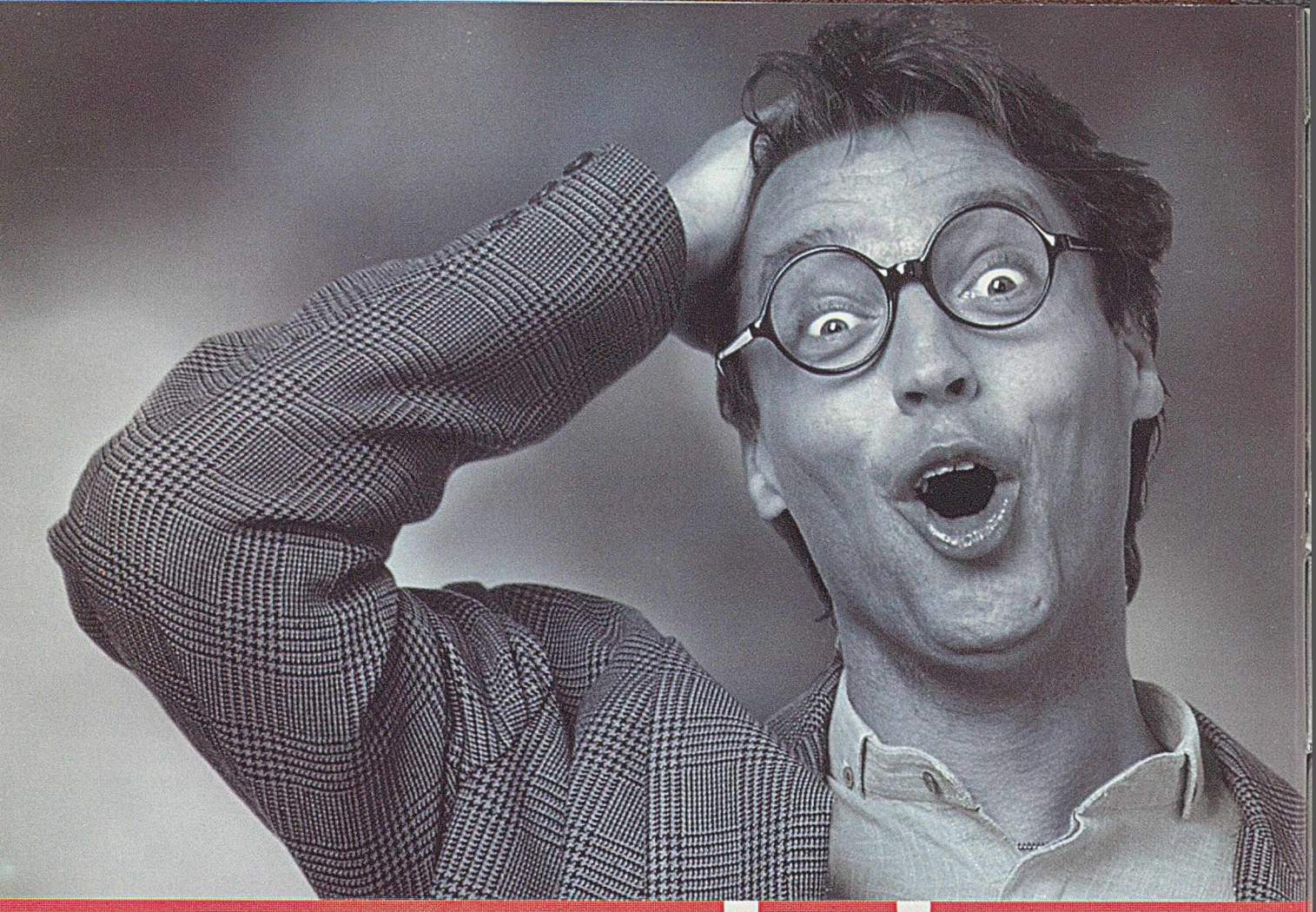
– Jakiej wiedzy może oczekiwać od absolwenta Wydziału Matematyki, Informatyki i Mechaniki jego pracodawca?

– To powinna być osoba, która posiadła znajomość kanonów wyższej matematyki i informatyki, umie rozwią-

dokończenie na s. 1

WYDAWNICTWO SIGMA NOT

Cena 5,00 zł (50 000 zł)



Aż tak dobra?!

OKIPAGE 4w

drukarka stronicowa

Tak - i możesz ją mieć!

Tak dobra to mało powiedziane, bo przecież drukuje z rozdzielczością 600 dpi i jakością lasera (ostre kontury, łagodne łuki, szeroka skala szarości). Czy to tekst czy grafika wypuszcza 4 strony na minutę na papierach różnych formatów. **Tak dobra** bo przeznaczona do pracy w środowisku Windows 3.1 i Windows 95. Tak dobra, tak szybka, tak prosta w obsłudze **za 999 zł - i możesz ją mieć.**

O/8/96

OKI

People to People Technology

P.1847/96
POLITECHNIKA ŚLĄSKA Biblioteka Główna w Gliwicach

Od teorii do praktyki

dokończenie z pierwszej strony okładki

zywać problemy, ma zdolność szybkiego uczenia się nowych rzeczy oraz zna różne stosowane na świecie nowoczesne metody i narzędzia (aczkolwiek nie musi mieć ich opanowanych z nadzwyczajną zręcznością).

– Czy nasz dynamicznie rozwijający się rynek informatyczny wpływa na programy nauczania informatyki?

– Tak, ale pamiętajmy, że Uniwersytet musi nauczać przede wszystkim podstaw, pewnych niezmiennych zasad oraz rozwijać naukę. Informatyka z racji swego młodego wieku bardzo się zmienia i pewne problemy, które przed dwudziestu laty mogły być przedmiotem badań doktorskich, przed dziesięcioma laty starczyłyby może na magisterium, teraz bywają omawiane na ćwiczeniach czy laboratoriach. Określenie tego, co jest w informatyce trwałe, co się nie zdeaktualizuje, nie jest rzeczą łatwą. Staramy się jednak modernizować programy nauczania tak, aby z jednej strony zachować ów uniwersytecki charakter studiów, a z drugiej dobrze przygotować naszych absolwentów do pracy w pozaakademickim środowisku. Jednakże nie zawsze możemy zaoferować atrakcyjne zajęcia, do niektórych brakuje nam specjalistów. Trudno na przykład oczekiwać od kolegi, który zajmuje się teorią typów, by poprowadził dobry wykład z sieci komputerowych. Zatrzymanie na uczelni kogoś, kto zna się dobrze na nowoczesnych, praktycznych działach informatyki, jest bardzo trudne, bo takie osoby zarabiają wielokrotnie więcej w firmach komputerowych. Ponadto, wyniki naukowe trudniej uzyskuje się w dziedzinach, które wiążą się z praktycznym zastosowaniem, a od takich wyników zależy przecież kariera akademicka. Poza modernizacją samych programów nauczania, a dotyczy to nie tylko informatyki, ale może nawet w większym stopniu matematyki (bo popyt na „czystych" matematyków jest coraz mniejszy), unowocześniamy także strukturę i tok studiów. Część osób będzie mogła zakończyć naukę po trzech latach, bez magisterium, z tytułem licencjata. Oferujemy różne potoki studiów, pozwalające zdobywać przydatne kwalifikacje w wybranych zastosowaniach, np. w matematyce finansowej. Od czterech lat prowadzimy także płatne trzyletnie zawodowe studia informatyczne, z oddzielnym programem, w którym jest mniej teorii, a więcej praktyki i ćwiczeń w laboratoriach. Staraliśmy się jednak, aby ten program dawał szansę kontynuacji studiów na poziomie magisterskim i część absolwentów ZSI z tego korzysta. Choć więc z racji naszej wspólnoty z matematykami studia informatyczne na naszym Wydziale dają świetne przygotowanie teoretyczne i metodyczne, dbamy troskliwie także i o tych, którzy nie mając na celu kariery akademickiej są bardzo potrzebni w kraju na różnorodnych stanowiskach pracy. Patrząc na popularność naszego Wydziału sądzę, że udaje się nam to nieźle robić.

– Czy wielu studentów jest gotowych zrezygnować z tytułu magistra na rzecz wcześniej podjętej pracy?

– Tak, i nic dziwnego, skoro i bez tytułu mogą uzyskać atrakcyjną pracę oraz uposażenie przewyższające pensję profesora z 30-letnim stażem.

– Polscy informatycy są bardzo cenieni za granicą. Ale w kraju narzeka się na brak takich specjalistów, jak na przykład integrator systemu.

– Na to składa się kilka przyczyn. Najważniejsza to wspomniana już żenująca sytuacja finansowa nauczycieli akademickich. Trudno jest zatrzymać na uczelni młodych, zdolnych ludzi zainteresowanych popularnymi praktycznymi działami informatyki, jak sieci komputerowe czy bazy danych. Ci, którzy się na tym znają, mogą sprzedać swoją wiedzę za duże pieniądze. Dlatego też pewnych kierunków nie możemy rozwijać w stopniu wystarczającym. Ale chciałbym jeszcze raz obalić mit, że na Uniwersytecie kształci się tylko „teoretyków". Warto może przypomnieć, że to ekipa studentów naszego Wydziału wygrywając jesienią 1994 roku w Amsterdamie eliminacje, awansowała do światowych finałów prestiżowego studenckiego konkursu programistycznego, organizowanego przez amerykańskie towarzystwo naukowe ACM (jako pierwsza polska drużyna w dziewięnastoletniej historii tych konkursów). Założeniem konkursów ACM jest zespołowa praca i nie tylko znalezienie algorytmów, ale i bezbłędne ich zakodowanie w zadanym środowisku. Jesienią 1995 roku powtórzyliśmy ten sukces, wygrywając eliminacje tym razem w Bratysławie i z 1001 drużyn awansowaliśmy do 43 zespołowej czołówki światowej, znowu jako jedyny zespół z Polski (Europę reprezentowało 10 drużyn, 5 było z Dalekiego Wschodu, jedna z Australii, jedna z Nowej Zelandii, a reszta to Ameryka Północna).

– Czy w nauczaniu informatyki nie dałoby się wprowadzić w życie lansowanego u nas bez efektów hasła: więź nauki z przemysłem? To rozwiązałoby problemy finansowe nauczycieli akademickich.

– Tak, ale brak jest mechanizmów, które by to umożliwiły. Na Zachodzie o zweryfikowanie znaczącego systemu informatycznego prosi się naukowców; sam brałem w Kanadzie udział w końcowej fazie prac mających na celu sprawdzenie poprawności oprogramowania sterującego awaryjnym wyłączaniem elektrowni atomowej. Przy wykonywaniu takiego zadania naukowcy zdobywają dodatkowe doświadczenie (i niebanalne korzyści finansowe), ich badania są inspirowane ważnymi zastosowaniami, a wyniki prac są publikowane w dobrych czasopismach. Podejmowaliśmy dawniej próby takiej współpracy, m.in. z przemysłem chemicznym, mieliśmy swego czasu kontakt z Żarnowcem, negocjowaliśmy z FSM, ale nic sensownego z tego nie wyszło. Także obecnie prowadzimy rozmowy z różnymi krajowymi i zagranicznymi instytucjami oraz firmami, ale nawiązanie długoterminowej współpracy gwarantującej uczestniczącym osobom i godziwe warunki finansowe i możliwość wykazania się publikowalnymi wynikami, nie jest łatwe. A przy tym trudno jest znaleźć entuzjastów do niepewnego przedsięwzięcia, gotowych do szybkiego działania – wiadomo bowiem powszechnie, że każdy pracownik naukowy, by utrzymać rodzinę jest uwikłany w tysiące różnych prac, których nie zostawi dla niepewnej doraźnej korzyści. Kiedy powstaną mechanizmy pozwalające na podejmowanie współpracy z firmami, to będzie to z obopólną korzyścią i w szczególności wspomoże nauczanie praktycznych dziedzin informatyki.

– Dziękuję Panu za rozmowę.

publikacje

Zastosowanie komputera do automatycznego układania rozkładu zajęć dla szkoły wyższej

Beata Jendrzejczyk
Marek Skomorowski
Instytut Informatyki
Uniwersytet Jagielloński
Kraków

Zastosowanie komputera do układania rozkładu zajęć jest tematem wielu publikacji. Na przykład, zamieszczona w [5] literatura liczy około 300 pozycji dotyczących zastosowania komputera do układania rozkładu zajęć. W typowym problemie układania rozkładu zajęć w szkole wyższej występują uczestnicy w sensie ogólnym, tj. studenci, sale wykładowe, laboratoria itd. Występuje również zbiór godzin (jednostek lekcyjnych). Dostępność określa podzbiór godzin, w których każdy z uczestników może brać udział w zajęciach, do których został przypisany. Każde spotkanie jest opisane przez zbiór wszystkich uczestników przypisanych do danych zajęć oraz liczbę godzin określających czas ich trwania. Rozkład zajęć jest harmonogramem przypisującym wszystkim przewidzianym spotkaniom godziny ich przeprowadzenia, spełniającym jednocześnie podstawowe żądanie: żaden z uczestników nie może być zaplanowany więcej niż jeden raz o tej samej godzinie. Powyżej został przedstawiony problem podstawowy. W praktyce, w zależności od typu szkoły wyższej, istnieje duża różnorodność żądań specjalnych, które rozkład zajęć powinien uwzględniać.

Problem układania rozkładu zajęć jest problemem NP-zupełnym ([1-2]). Oznacza to, że nie istnieje efektywny, to znaczy o wielomianowej złożoności obliczeniowej algorytm konstruowania optymalnego, ze względu na przyjęty wskaźnik jakości, rozkładu zajęć w przypadku ogólnym.

Prezentowany tu komputerowy system automatycznego układania rozkładu zajęć jest oparty na algorytmie heurystycznym, przedstawionym w [4, 6], którego działanie można opisać ogólnie w następujący sposób.

Na podstawie danych wejściowych jest przygotowywana lista zajęć do zaplanowania. Każde zajęcia mają określony priorytet. Priorytety zajęć są wyliczane na podstawie analizy danych wejściowych. Następnie algorytm próbuje ułożyć kolejno wszystkie zajęcia z listy zajęć do zaplanowania, rozpoczynając od zajęć o najwyższym priorytecie, spełniając jednocześnie ograniczenia podane w danych wejściowych. Zajęcia ułożone są usuwane z listy zajęć do zaplanowania. W przypadku ułożenia wszystkich zajęć algorytm kończy pracę. W przeciwnym przypadku priorytety zajęć nie ułożonych są zwiększa-

```
Rozkład zajęć( )
 {
  WCZYTAJ_dane( );
  PRZYPISZ_zajęciom_priorytety( );
  iteracja = 1;
  wskaźnik_jakości = 1;
  while  ( ( wskaźnik_jakości > 0 )  i  ( iteracja  <  MAXITERACJI ) )
   {
    POSORTUJ_zajęcia_według_przypisanych_im_priorytetów( );
    wskaźnik_jakości  =  UŁÓŻ_ROZKŁAD_( );
                                 /* funkcja zwraca 0 (w przypadku
                                 /* ułożenia wszystkich zajęć) lub
                                 /* liczbę godzin nie ułożonych
    if ( wskaźnik_jakości  >  0 )
     {
      if ( iteracja = 1 )            /* rozkład zajęć układany pierwszy raz
       {
        ZAPAMIĘTAJ_ułożony_rozkład_zajęć( );
        ZAPAMIĘTAJ_listę_zajęć_nie_ułożonych( );
       }
      else
       {
        PORÓWNAJ_z_najlepszym_uzyskanym_dotychczas_rozkładem_zajęć( );
        if ( ułożony_w_bieżącej_iteracji_rozkład_zajęć_jest_lepszy )
         {
          ZAPAMIĘTAJ_ułożony_rozkład_zajęć( );
          ZAPAMIĘTAJ_listę_zajęć_nie_ułożonych( );
         }
       } /* else
      ZWIĘKSZ_PRIORYTETY_zajęć_nie_ułożonych( );
     } /* if
   } /* while
  WYPISZ_NAJLEPSZY_ułożony_rozkład_zajęć( );
  if ( lista_zajęć_nie_ułożonych_nie_jest_pusta )
   WYPISZ_listę_zajęć_nie_ułożonych( );
 } /* Rozkład zajęć */
```

Rys. 1. Algorytm układania rozkładu zajęć

ne i przygotowywana jest lista zajęć do zaplanowania, uwzględniająca wyższe priorytety zajęć poprzednio nie ułożonych. Następnie algorytm próbuje kolejny raz ułożyć zajęcia z listy zajęć do zaplanowania. Poszukiwanie kolejnych rozwiązań jest kontynuowane do momentu ułożenia wszystkich zajęć z listy zajęć do zaplanowania lub do momentu uzyskania rozwiązania najlepszego, ze względu na przyjęty wskaźnik jakości, w ramach zadanej liczby prób (kolejnych iteracji algorytmu). Jako wskaźnik jakości została przyjęta liczba godzin nie ułożonych w danym rozkładzie. W dowolnym momencie obliczeń są pamiętane co najwyżej dwa rozkłady zajęć: układany w danej iteracji algorytmu i najlepszy spośród rozkładów uzyskanych w poprzednich iteracjach. Po uzyskaniu rozwiązania końcowego są wyprowadzane rozkłady dla poszczególnych zajęć, wykładowców i sal. W przypadku nie ułożenia wszystkich zajęć jest wyprowadzany również zbiór zajęć nie zaplanowanych. Omówiony algorytm jest pokazany na rysunku 1.

Dane wejściowe systemu są zapisywane w trzech plikach zawierających informacje o zajęciach, salach i wykładowcach. Fragment przykładowego pliku zawierającego informacje o zajęciach jest pokazany na rys. 2.

```
#
1. Uklady cyfrowe
2. UC
3. I

01. dr M. Skomorowski
02. 40
03. 2

11. dr M. Skomorowski
12. 20
13. 2

21. mgr H. Telega
22. 20
23. 2
#
```

Rys. 2. Fragment przykładowego pliku zawierającego informacje o zajęciach

Fragment ten zawiera informacje na temat wykładu zatytułowanego „Układy cyfrowe" (pole 1) o kodzie UC (pole 2), należącego do grupy przedmiotów informatycznych (kod I w polu 3). Wykład ten jest planowany dla 40 osób (pole 02) z dwiema grupami ćwiczeniowymi (pola 11, 21) dla 20 osób (pola 12, 22). Wykład i ćwiczenia będą się odbywać raz w tygodniu (pole 03 – wykład, pola 13, 23 – ćwiczenia). Podane są również nazwiska prowadzących wykład (pole 01) i ćwiczenia (pola 11, 21). Znak # jest separatorem poszczególnych zajęć. W pliku tym można podać również inne informacje. Na przykład: wymagania dotyczące terminu i sali dla danych zajęć, wymagania dotyczące tego, ile razy w tygodniu i po ile godzin powinny odbywać się dane zajęcia. W przypadku kiedy zajęcia z tego samego przedmiotu odbywają się dwa razy w tygodniu, można podać, jak długa ma być przerwa między nimi.

```
#
221
50
#
217 p
16
zajeta czw 16.00 - 20.00, pt 8 - 16
#
218 p
16
#
```

Rys. 3. Fragment przykładowego pliku zawierającego informacje o salach

Fragment przykładowego pliku zawierającego informacje o salach jest pokazany na rys. 3. Fragment ten zawiera informacje o trzech salach (221, 217, 218). Sala 221 może pomieścić 50 studentów. Sale 217 i 218 są pracowniami komputerowymi *(p)* mogącymi pomieścić 16 osób. Sale 221 i 218 są dostępne w ciągu całego tygodnia. Sala 217 jest zajęta w czwartki w godzinach od 16:00 do 20:00 i w piątki w godzinach od 8:00 do 16:00. Znak # jest separatorem poszczególnych sal.

```
#
dr M. Skomorowski
1. pn 8.00 - 20.00, pt 8.00 - 10.00
2. wt, sr, czw 8.00 - 20.00
#
```

Rys. 4. Fragment przykładowego pliku zawierający informacje o prowadzących zajęcia

Fragment przykładowego pliku zawierającego informację o dostępności prowadzących zajęcia jest pokazany na rys. 4. Zawiera informację o ograniczeniach czasowych prowadzącego zajęcia. Pole 1 zawiera informacje o terminach, w których prowadzący zajęcia nie jest dostępny, a pole 2 o terminach preferowanych przez prowadzącego zajęcia.

```
Uklady cyfrowe  UC
wyklad: dr M. Skomorowski  sr 14 - 16  s 221
grupy cwiczeniowe:
gr.1  dr M. Skomorowski  sr 12 - 14  s 221
gr.2  mgr H. Telega  sr 12 - 14  s 205
```

Rys. 5. Fragment przykładowego rozkładu dla zajęć

Godzina	Poniedziałek	Wtorek	Środa	Czwartek	Piątek
8 - 9	MD2 w	WDM w	LSK w		
9 - 10	MD2 w	WDM w	LSK w		
10 - 11	RPS w	MT w	WDI w		AL1 gr.3
11 - 12	RPS w	MT w	WDI w		AL1 gr.3
12 - 13	RPS w	WDI w	UC gr.1	MP2 gr.3	AL1 gr.3
13 - 14		WDI w	UC gr.1	MP2 gr.3	
14 - 15	AL1 w	MP2 w	UC w		
15 - 16	AL1 w	MP2 w	UC w		
16 - 17	AL1 w	TP gr.1	WZTW w		
17 - 18	MNA w	TP gr.1	WZTW w		
18 - 19	MNA w	TP w			
19 - 20	NI w	TP w			

Rys. 6. Fragment przykładowego rozkładu dla sal sala 221

dokończenie na s. 14

Wprowadzenie do systemów uczących się ze wzmocnieniem

Paweł Cichosz
Instytut Podstaw Elektroniki
Politechnika Warszawska

Artykuł rekomendowany

Artykuł ma na celu przybliżenie Czytelnikowi mało znanego w Polsce, a bez wątpienia godnego uwagi i budzącego gwałtownie rosnące w ostatnich latach zainteresowanie na świecie, działu uczenia się maszyn, nazywanego uczeniem się ze wzmocnieniem (ang. *reinforcement learning*). Jest to jeden z najnowszych podobszarów uczenia się maszyn, przeżywający w ostatnich latach prawdziwą eksplozję wzrostu zainteresowania i zaliczany do tzw. „gorących tematów", zwłaszcza w USA, natomiast bardzo słabo znany w Polsce. Czytelników zainteresowanych bardziej wszechstronną dyskusją różnych metod uczenia się maszyn odsyłamy do prac [5, 6, 16].

Rozważa się w nim problemy, w których uczący się system ma zadanie nauczenia się podejmowania decyzji w pewnym nieznanym środowisku, na podstawie skalarnych rzeczywistoliczbowych wartości nagrody lub kary. Koncentrować się będziemy na najbardziej interesującym przypadku, w którym owe nagrody lub kary mogą być opóźnione w stosunku do akcji systemu, których są konsekwencją. Po scharakteryzowaniu paradygmatu uczenia się ze wzmocnieniem i jego miejsca w dziedzinie uczenia się maszyn, rozważymy kluczowy dla jego realizacji praktycznej problem temporalnego przypisania zasługi. Omówiony zostanie najbardziej obecnie znany i najczęściej stosowany algorytm rozwiązywania tego problemu, Q-learning. Artykuł zamknie przegląd najbardziej obiecujących obszarów zastosowań systemów uczących się ze wzmocnieniem.

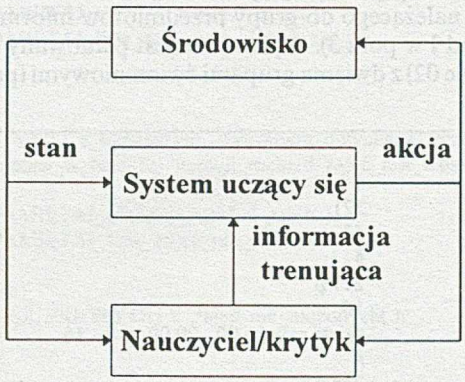
Uczenie się w sztucznej inteligencji

Wysiłki badawcze, podejmowane już od ponad czterdziestu lat w dziedzinie *sztucznej inteligencji* (SI), mają na celu wypracowanie metod, dzięki którym możliwe byłoby konstruowanie sztucznych systemów (komputerowych) zdolnych do wykonywania zadań, o których sądzi się, że wymagają inteligencji pokrewnej (w każdym razie pod pewnymi względami) inteligencji ludzkiej. Takie, tyleż ambitne co nieprecyzyjne określenie tej dziedziny, było i wciąż bywa przyczyną wielu kontrowersji, dotyczących zarówno tego, co do niej należy, a co nie, jak i tego, czy w ogóle może ona rościć sobie prawa do miana dyscypliny naukowej.

Elementem inteligentnego zachowania, którego kluczowa rola jest najbardziej chyba bezdyskusyjna, jest zdolność do uczenia się. Spośród wielu możliwych definicji uczenia się, podawanych zarówno w kontekście uczenia się ludzi, zwierząt, jak i maszyn, wybierzmy najprostszą: *Uczenie się jest procesem poprawy jakości działania systemu na podstawie doświadczeń z przeszłości.* Z kontekstu związanego z konkretnie rozważanym procesem uczenia się wynika, co dokładnie rozumie się przez poprawę jakości działania i jaki rodzaj doświadczeń może być wykorzystywany do jej osiągnięcia. Zawsze można jednak przyjąć, że jej niezbędnym warunkiem jest zdobywanie i doskonalenie wiedzy lub umiejętności oraz odpowiednie ich wykorzystywanie w dalszych działaniach systemu.

Paradygmaty uczenia się

W dalszej dyskusji posłużymy się ogólnym schematem interakcji systemu uczącego się z jego środowiskiem przedstawionym na rys. 1. Bez istotnego zmniejszania ogólności rozważań można przyjąć, że interakcja ta polega na obserwowaniu przez system uczący się aktualnego *stanu* środowiska oraz generowaniu w odpowiedzi



Rys. 1. Ogólny schemat interakcji systemu uczącego się ze środowiskiem

Rekomenduje:
Prof. dr hab. Jan J. Mulawka
Instytut Podstaw Elektroniki
Politechnika Warszawska

Praca wspierana przez KBN w ramach grantu nr 8 S503 019 05.

na ten stan pewnej *akcji* na podstawie jego aktualnej wiedzy. Systemowi może być dostarczona informacja trenująca, której źródło będziemy, w zależności od jej charakteru, nazywać *nauczycielem* lub *krytykiem*. Informacja ta może, w ogólnym przypadku, zależeć zarówno od stanu środowiska, jak i od wykonanej akcji systemu. Na podstawie interakcji ze środowiskiem z jednej strony oraz informacji trenującej z drugiej strony, system może doskonalić swoją wiedzę lub umiejętności.

Uczenie się bez nadzoru

W przypadku uczenia się bez nadzoru (ang. *unsupervised learning*) celem jest zazwyczaj przetransformowanie informacji wejściowej, dostarczonej ze środowiska, do pewnych innych form lepiej dopasowanych do dalszego przetwarzania. Typowym przykładem może być zadanie grupowania (ang. *clustering*), w którym porcje danych wejściowych (w naszej terminologii reprezentowanych przez stany środowiska) muszą być podzielone na pewne rozłączne klasy lub kategorie. Może to dotyczyć zarówno punktów w pewnych przestrzeniach rzeczywistoliczbowych, jak i symbolicznych opisów pewnych obiektów, w tym ostatnim przypadku mamy do czynienia z tzw. grupowaniem pojęciowym (ang. *conceptual clustering*).

Zasada, według której następuje transformowanie danych wejściowych w uczeniu się bez nadzoru, jest ustalona dla określonej klasy zadań, które rozwiązuje system uczący się, i wbudowana do jego wewnętrznej struktury. Nie ma żadnego źródła zewnętrznej informacji trenującej, które określałoby pożądany sposób działania systemu. W związku z tym mówi się niekiedy, że systemy uczące się bez nadzoru mają wbudowanego nauczyciela lub krytyka.

Uczenie się z nadzorem

Uczenie się z nadzorem (ang. *supervised learning*) jest nazywane także uczeniem się z nauczycielem dla podkreślenia charakteru wykorzystywanej informacji trenującej. Pożądany sposób działania systemu jest w tym przypadku określany przez zewnętrzne źródło informacji trenującej za pomocą tzw. *wzorców trenujących* lub *przykładów*, w podstawowym wariancie rozumianych jako pary wiążące przykładowe wejście systemu (stan) z wymaganym dla niego wyjściem (akcją).

Paradygmat uczenia się z nadzorem obejmuje wyraźną większość metod uczenia się maszyn i wiele różnorodnych problemów, takich jak uczenie się opisów pojęć i reguł decyzyjnych z przykładów, indukcja drzew decyzyjnych, indukcyjne programowanie logiczne, aproksymacja funkcji i inne. W praktyce informacja dostarczana przez nauczyciela może przybierać różne formy. W przypadku uczenia się pojęć są to przykłady pozytywne, które musi obejmować opis pojęcia konstruowany przez system, i negatywne, których ten opis obejmować nie może. W przypadku uczenia się aproksymacji funkcji każda para trenująca może podawać wymagane wyjście dla danego wejścia, wektor błędu – różnicy pomiędzy docelowym a aktualnym wyjściem systemu, bądź też tylko informację gradientową, określającą kierunek w przestrzeni wyjściowej, w którym powinna ulec zmianie odpowiedź systemu dla danego wejścia. Mimo tej różnorodności, w każdym problemie uczenia się z nadzorem, informacja trenująca ma charakter dyrektywny i specyfikuje bezpośrednio lub pośrednio pożądany sposób zmiany działania systemu.

Uczenie się ze wzmocnieniem

W przypadku *uczenia się ze wzmocnieniem* (ang. *reinforcement learning*) również istnieje zewnętrzne źródło informacji trenującej, ma ono jednak naturę raczej krytyka niż nauczyciela. Informacja trenująca ma charakter *wartościujący* a nie dyrektywny. Zatem systemowi uczącemu się nie mówi się nigdy, jakich akcji oczekuje się od niego w określonych stanach, ani nawet jakie akcje (jeśli w ogóle jakiekolwiek) byłyby lepsze od tych, które on faktycznie wykonuje. Dostarczany jest natomiast skalarny rzeczywistoliczbowy sygnał *wzmocnienia*, nazywanego także wypłatą lub nagrodą (karą), stanowiący pewną względną miarę jakości działania systemu ze względu na stawiane przed nim cele. System musi samodzielnie ustalić, jakie akcje wykonywać w każdym stanie, aby maksymalizować uzyskiwane wartości wzmocnienia. Przyporządkowanie stanom akcji do wykonania, którego uczy się system, nazywać będziemy *strategią decyzyjną*. W przestrzeni możliwych strategii będziemy wyróżniać strategie optymalne, wyznaczane w pewien sposób, którego na razie nie precyzujemy, przez wartości nagród, do otrzymania których prowadzą.

Termin „wzmocnienie" wywodzi się z badań nad uczeniem się zwierząt w psychologii behawiorystycznej. Po raz pierwszy został on przypuszczalnie wprowadzony do informatyki przez Minsky'ego [15]. Podstawowa zdroworozsądkowa idea, jaka się za nim kryje, jest następująca: jeśli pewna akcja pociąga za sobą zadowalający stan rzeczy lub poprawę w stanie rzeczy, to tendencja do wykonywania tej akcji (w danej sytuacji) powinna zostać *wzmocniona* [1]. Jak zobaczymy w następnym podrozdziale, pojęcie „pociągania za sobą zadowalającego stanu rzeczy" może być rozciągnięte na długoterminowe konsekwencje akcji.

Uczenie się na podstawie opóźnionych nagród

Uczenie się ze wzmocnieniem, chociaż wyodrębnione przez nas najpierw z dziedziny sztucznej inteligencji, a potem z uczenia się maszyn, stanowi samo w sobie dziedzinę stosunkowo pojemną. Uściślijmy zatem, które z obejmowanych przez nią zagadnień uważamy za najbardziej interesujące, i na których chcemy się dalej skoncentrować.

Podstawowy scenariusz

Zaczniemy od kilku oznaczeń i uzupełnienia terminologii. Będziemy używać symboli x_t, a_t i r_t do oznaczenia, odpowiednio, stanu, akcji i wartości wzmocnienia w chwili czasu t. Czwórkę złożoną ze stanu, akcji, otrzymanego po jej wykonaniu w tym stanie wzmocnienia oraz następnego stanu nazywać będziemy *doświadczeniem systemu*. Przy użyciu wprowadzonych właśnie symboli doświadczenie z kroku czasu t zapiszemy jako $<x_t, a_t, r_t, x_{t+1}>$. Podstawowy scenariusz uczenia się ze wzmocnieniem można wówczas przedstawić jako sekwencję operacji

wykonywanych przez system uczący się w każdym kroku dyskretnego czasu t (rys.2).

W każdym kroku czasu t:
1. obserwuj aktualny stan x_t;
2. wybierz akcję a_t do wykonania w stanie x_t zgodnie z aktualną strategią decyzyjną;
3. wykonaj akcję a_t;
4. obserwuj wartość wzmocnienia r_t i nowy stan x_{t+1};
5. aktualizuj strategię decyzyjną na podstawie doświadczenia $\langle x_t, a_t, r_t, x_{t+1}\rangle$.

Rys. 2. Podstawowy scenariusz uczenia się ze wzmocnieniem

Szerokie spektrum algorytmów uczenia się ze wzmocnieniem można przedstawić jako jego uściślone wersje, przy czym uściślenie dotyczy w gruncie rzeczy tylko jednego (ale za to kluczowego) kroku 5, w którym następuje aktualizacja strategii decyzyjnej pod wpływem kolejnego doświadczenia.

Środowisko

Środowisko jest pewnym złożonym systemem, którego stany, zmieniające się pod wpływem akcji systemu uczącego się, mogą być przezeń obserwowane. Odstępując chwilowo od modelu z rys. 1, założymy dla wygody, że także wartości wzmocnienia są dostarczane przez środowisko. Przyjmiemy, że zarówno zmiany stanów środowiska, jak i generowane przez nie nagrody mogą być, w ogólnym przypadku, stochastyczne. Będziemy też zakładać, że odpowiadające im rozkłady prawdopodobieństw (w pełni opisujące środowisko) nie są znane systemowi uczącemu się, ale że jednocześnie dostępna mu informacja wejściowa w postaci stanu, obserwowanego w każdym kroku, jest dostateczna do podjęcia optymalnej decyzji (czyli istotnie optymalna strategia może być reprezentowana jako odwzorowanie ze stanów na akcje). Powszechnie przyjęty matematyczny model dla problemu uczenia się ze wzmocnieniem w tego typu środowiskach stanowi *proces decyzyjny Markowa*, dla którego optymalne strategie decyzyjne mogą być odnalezione za pomocą metod stochastycznego programowania dynamicznego [4], pod warunkiem jednak, że znane są prawdopodobieństwa zmian stanów i oczekiwane wartości wzmocnienia dla każdej pary stan–akcja. W przypadku, gdy wiedza taka nie jest dostępna, otwiera się pole do stosowania metod uczenia się ze wzmocnieniem.

Miara optymalności strategii

Zadaniem systemu uczącego się ze wzmocnieniem jest nauczenie się optymalnej strategii decyzyjnej, to jest takiej, która prowadzi do optymalizacji pewnej miary jakości jego działania opartej na otrzymywanych wartościach wzmocnienia. W przypadku najbardziej interesującym i stanowiącym największe wyzwanie chodzi o maksymalizację nagród *w długim horyzoncie czasowym*. System ma więc wybierać swoje akcje tak, aby uzyskiwać wysokie dochody niekoniecznie natychmiast, lecz raczej długoterminowo. Tak jak w grze w szachy warto oddać piona po to, aby kilka ruchów później zagrozić królowi przeciwnika, może opłacać się wykonać kilka akcji nisko nagradzanych po to, aby po pewnej liczbie kroków stworzyć możliwość uzyskania wysokich wartości wzmocnienia. Ten rodzaj uczenia się jest nazywany *uczeniem się na podstawie opóźnionych nagród lub uczeniem się z opóźnionym wzmocnieniem* (ang. *learning from delayed rewards, delayed reinforcement learning*) [17, 22] i właśnie jemu chcemy poświęcić dalszy ciąg tego tekstu.

Zadanie długoterminowej maksymalizacji wzmocnienia można formalizować w różny sposób. Ograniczymy się tutaj do zdecydowanie dominującego obecnie, choć nie wolnego od pewnych niedostatków podejścia, w którym miarę jakości działania systemu określa się jako oczekiwaną całkowitą *zdyskontowaną* sumę wzmocnienia, jakie uzyskuje on w okresie swojego funkcjonowania, czyli:

$$\mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right],$$

gdzie E jest symbolem wartości oczekiwanej, r_t, jak wyżej, oznacza nagrodę otrzymaną w kroku czasu t, zaś $0 \leqslant \gamma \leqslant 1$ jest tzw. *współczynnikiem dyskontowania* (ang. *discount factor*), który determinuje stopień względnej ważności nagród bliskich i odległych w czasie. Jeśli $0<\gamma<1$, wartości wzmocnienia z kolejnych kroków czasu są ważone wykładniczo malejącymi współczynnikami, zgodnie ze zdroworozsądkową zasadą, że nagrody stają się mniej atrakcyjne (a kary mniej odstraszające), jeśli mają być otrzymane w odległej przyszłości. Aby maksymalizować powyższe wyrażenie dla dowolnego $\gamma>0$ system uczący się musi brać pod uwagę nie tylko natychmiastowe, ale także długoterminowe konsekwencje swoich akcji.

Realizacja uczenia się ze wzmocnieniem

Dla realizacji uczenia się ze wzmocnieniem kluczowe znaczenie ma problem *temporalnego przypisania zasługi* (ang. *temporal credit assignment*) [17]. Polega on na przypisaniu „zasługi" bądź „winy" za całościowe wyniki działania systemu (a więc jego długoterminowe nagrody) poszczególnym akcjom podjętym przez system, być może wykonanym wiele kroków wcześniej, zanim wyniki te mogły być zaobserwowane. Obecnie najwięcej uwagi poświęca się algorytmom rozwiązującym ten problem, które są oparte na metodach *różnic czasowych* (ang. *temporal differences*) Suttona [18], zaś najbardziej znanym z nich jest algorytm *Q-learning* [22, 23].

Algorytm Q-learning

Algorytm Q-learning konstruuje oszacowanie pewnej funkcji, nazywanej Q-funkcją, która wartościuje wszystkie pary stan–akcja ze względu na oczekiwane przyszłe nagrody. Dokładniej, każdej parze stan–akcja (x,a) przyporządkowuje ona tzw. Q-wartość $Q(x,a)$, która docelowo ma być oszacowaniem skumulowanego zdyskontowanego wzmocnienia, jakie będzie otrzymane po wykonaniu akcji a w stanie x, a następnie posługiwaniu się strategią *zachłanną* względem aktualnych Q-wartości (tj. wybieraniu w każdym kolejnym stanie akcji, której Q-wartość jest największa). Zatem, ponieważ w każdym stanie Q-funkcja zawiera (uzyskane na podstawie dotych-

czasowych doświadczeń) informacje na temat spodziewanych przyszłych długoterminowych skutków poszczególnych akcji, umożliwia ona podejmowanie optymalnych decyzji bez żadnego kosztownego planowania czy przeszukiwania w przód przestrzeni stanów. W każdym kroku Q-funkcja jest modyfikowana zgodnie z algorytmem przedstawionym na rys. 3. Występujące tam odwołania do Q-wartości zawierają indeks t określający krok czasu, z którego wartości te pochodzą.

1. W każdym kroku czasu t:
2. obserwuj aktualny stan x_t;
3. wybierz akcję a_t do wykonania w stanie x_t (na podstawie $Q(x_t,a_t)$ dla każdego a);
4. wykonaj akcję a_t;
5. obserwuj wartość wzmocnienia r_t i nowy stan x_{t+1};
6. $\Delta := r_t + \gamma \max_a Q_t(x_{t+1},a) - Q_t(x_t,a_t)$;
7. $uaktualnij^{\beta}(Q, x_t, a_t, \Delta)$;

Rys. 3. Algorytm Q-learning

Wybór akcji. W kroku 2 następuje wybór akcji do wykonania w aktualnym stanie, na podstawie wartości Q-funkcji dla tego stanu. Jest naturalne, że przy wyborze tym należy preferować akcje o najwyższych Q-wartościach (skoro po ich wykonaniu oczekiwane są najwyższe przyszłe nagrody), nie może to być jednak wybór w pełni deterministyczny – wykonywanie zawsze akcji, które obecnie wydają się najlepsze, może uniemożliwić systemowi uczącemu się odkrycie innych, być może lepszych akcji. Mamy tu do czynienia z tzw. wymianą pomiędzy eksploracją (wykonywaniem akcji nie uważanych za najlepsze w celu poprawienia posiadanej wiedzy) i eksploatacją (korzystaniem z aktualnie dostępnej wiedzy w celu zdobywania nagród).

Najprostsze podejście do problemu wyboru akcji zapewniającego dostateczny poziom eksploracji polega na zastosowaniu pewnego stochastycznego mechanizmu wyboru. Typowym przykładem może być mechanizm wykorzystujący rozkład Boltzmanna, zgodnie z którym prawdopodobieństwo wyboru akcji a^* w stanie x jest równe:

$$\mathrm{Prob}(x,a^*) = \frac{\exp\left(Q(x,a^*)\ T\right)}{\sum_a \exp\left(Q(x,a)\ T\right)},$$

gdzie parametr T, nazywany *temperaturą*, jest liczbą dodatnią regulującą stopień losowości wyboru. Stosując duże wartości T uzyskujemy system wybierający akcje w sposób niemal losowy, zaś małe wartości powodują prawie deterministyczny wybór akcji o maksymalnych Q-wartościach.

Obliczenie błędu. Centralnym krokiem algorytmu jest krok 5, w którym jest obliczany błąd, używany następnie do zmiany Q-wartości dla aktualnego stanu i akcji. Jak powiedziano wyżej, $Q(x, a)$ ma być oceną oczekiwanej zdyskontowanej sumy wzmocnienia otrzymywanego po wykonaniu akcji a w stanie x i następnie zawsze wybieraniu akcji o maksymalnych Q-wartościach. Wówczas o $\max_a Q(x, a)$ można myśleć jako o pewnej mierze wartości stanu x (i pisać w zamian $V(x)$), oceniającej maksymalne całkowite zdyskontowane wzmocnienie, jakie może być otrzymane przez system rozpoczynający działanie w tym stanie. Stosowana przez algorytm Q-learning reguła obliczania błędu powoduje przybliżenie Q-wartości dla aktualnego stanu i aktualnej akcji do sumy natychmiastowego wzmocnienia otrzymanego po wykonaniu tej akcji oraz zdyskontowanej sumy wartości następnego stanu:

$$Q(x_t,a_t) \leftarrow r_t + \gamma V(x_{t+1}).$$

Uzasadnienie dla tej reguły wywodzi się z teorii programowania dynamicznego i nie będziemy go tutaj przytaczać, odsyłając Czytelników do oryginalnej pracy Watkinsa [22]. Poprzestaniemy na odnotowaniu, że pod pewnymi warunkami prowadzi ona do zbieżności Q-funkcji do optymalnej Q-funkcji, tj. takiej, że strategia zachłanna względem niej jest strategią optymalną. Ściśle sformułowane twierdzenie o zbieżności algorytmu Q-learning i jego dowód można znaleźć w [23].

Reprezentacja funkcji. Krok 6 algorytmu z rys. 3 wykorzystuje wyznaczony w poprzednim kroku błąd do aktualizacji Q-wartości dla aktualnego stanu i aktualnej akcji. Operacja ta, zapisana jako

$$uaktualnij^{\beta}(Q,x_t,a_t,\Delta),$$

ma za zadanie zmianę wartości $Q(x_t, a_t)$ w ten sposób, aby stała się ona bliższa $Q(x_t, a_t)+\Delta$, w stopniu kontrolowanym przez współczynnik szybkości uczenia β. Implementacja tej operacji zależy w oczywisty sposób od przyjętej reprezentacji funkcji Q.

Najprostsze podejście polega na przechowywaniu Q-wartości w tablicy, zawierającej jeden element dla każdej pary stan–akcja. Wówczas operacja aktualizacji, opisana przez równanie, implementowana jest w naturalny sposób jako przypisanie:

$$Q(x_t,a_t) := Q(x_t,a_t) + \beta\Delta.$$

Dostępne rezultaty teoretyczne dotyczące gwarancji zbieżności algorytmu Q-learning dotyczą tego właśnie, najprostszego przypadku [23]. Niestety, reprezentacja tablicowa ma poważne wady, zwłaszcza dla bardziej złożonych i realistycznych problemów. Po pierwsze, jej zapotrzebowanie na pamięć, proporcjonalne do liczby stanów i do liczby akcji, może okazać się nieakceptowalne. Po drugie, nie pozwala ona na uogólnianie w zbiorach podobnych stanów. W związku z tym były prowadzone liczne, głównie eksperymentalne prace dotyczące wykorzystywania przez algorytmy uczenia się ze wzmocnieniem innych metod reprezentacji funkcji, na ogół uczących się i generalizujących aproksymatorów funkcji [8, 12, 21, 24].

Metody TD(λ)

Jak już wspomniano wyżej, algorytm Q-learning jest przykładem algorytmów wykorzystujących metody różnic czasowych do temporalnego przypisania zasługi. W istocie jest to pewna klasa metod, parametryzowana przez tzw. *współczynnik świeżości* (ang. *recency factor*) $0 \leqslant \lambda \leqslant 1$, oznaczana jako TD ($\lambda$), przy czym Q-learning odpowiada najprostszemu przypadkowi TD (0). Metody te mogą być w ogólnym przypadku stosowane do uczenia się predykcji w wieloetapowych problemach predykcyj-

nych. W takich problemach na każdym etapie należy podać prognozę pewnej końcowej wielkości, która staje się dostępna po ostatnim etapie, jednak pewna cząstkowa informacja na jej temat dostępna jest na każdym etapie jako aktualny stan. Metody TD wykorzystują do uczenia się różnice predykcji z dwóch lub, dla $\lambda > 0$, z większej liczby kolejnych kroków czasu. Formalną definicję metod TD (λ) w ich ogólnej postaci można znaleźć w oryginalnej pracy Suttona [18]. Sposób wykorzystania tych metod do uczenia się ze wzmocnieniem jest obszernie dyskutowany w [7].

Zastosowania uczenia się ze wzmocnieniem

Paradygmat uczenia się ze wzmocnieniem jest sformułowany w sposób bardzo ogólny i abstrakcyjny. Czyni go to niezwykle pojemnym i szeroko stosowalnym. W istocie, aby sformułować problem w kategoriach uczenia się ze wzmocnieniem należy jedynie określić odpowiednio reprezentację stanów obserwowanych przez system na wejściu i rodzaj akcji generowanych przez niego na wyjściu oraz mechanizm wzmocnienia, stanowiący właściwą specyfikację zadania do wykonania. Przy pewnych zastrzeżeniach, można zaryzykować porównanie pierwszej z tych czynności do projektowania struktur danych, drugiej zaś do projektowania algorytmu w inżynierii programowania. Ponieważ rozważamy uczenie się na podstawie opóźnionych nagród, projektant, który za pomocą funkcji wzmocnienia specyfikuje systemowi zadanie do wykonania, nie musi wiedzieć dokładnie, kiedy poszczególne akcje zasługują na pozytywne lub negatywne wzmocnienie. Wystarczy, jeśli będzie potrafił wyróżnić i nagrodzić sytuacje szczególnie korzystne (takie, jak np. osiągnięcie pewnych cząstkowych celów zadania) oraz wyróżnić i ukarać sytuacje szczególnie niekorzystne (takie, jak np. popełnienie pewnych ewidentnych błędów).

Oczywiście, stosowanie metod uczenia się ze wzmocnieniem, chociaż często możliwe, nie zawsze jest sensowne. Jako generalną zasadę można przyjąć, że gdziekolwiek jest dostępna wiedza na temat środowiska i zadania systemu uczącego się pozwalająca na zastosowanie metod uczenia się z nadzorem, należy z niej skorzystać. Główną domeną metod uczenia się na podstawie nagród są dziedziny, w których taka wiedza nie jest dostępna, jest trudna do otrzymania lub sformułowania, bądź jej otrzymanie wiąże się z dużym kosztem (w sensie wysiłku ludzi lub dodatkowych obliczeń).

Do dziedzin, w których stosowanie metod uczenia się ze wzmocnieniem jest uzasadnione i było już intensywnie badane, należy na pewno automatyczne sterowanie. W uproszczeniu, w problemach sterowania rozważa się dynamiczny system, na którego wejście należy podawać odpowiednie sygnały sterowania tak, aby osiągnąć pewien predefiniowany cel. W przypadku, gdy cel ten jest sformułowany jako optymalizacja pewnej miary aktualnej sytuacji sterowanego systemu, mamy do czynienia z problemem optymalnego sterowania. Niekiedy odpowiedni sterownik można zbudować posługując się matematycznym opisem problemu. Jeśli jednak nie jest to możliwe lub jest możliwe tylko częściowo i przynajmniej niektóre parametry sterownika muszą być modyfikowane na podstawie obserwacji rzeczywistego zachowania się sterowanego systemu, mamy do czynienia ze sterowaniem adaptacyjnym.

Można, generalnie rzecz biorąc, rozróżnić dwie klasy podejść do jego realizacji. Podejście pośrednie polega na przeprowadzeniu wstępnie tzw. *identyfikacji systemu*, czyli skonstruowaniu jego wiarygodnego modelu, poprzez obserwację jego reakcji na podawane na jego wejście sygnały, następnie zaś określeniu strategii sterowania na podstawie tego modelu. W przypadku, gdy problem sterowania można reprezentować za pomocą procesu decyzyjnego Markowa, oznacza to najpierw identyfikację prawdopodobieństw zmian stanów i wartości oczekiwanych nagrody, a następnie znalezienie optymalnej strategii sterowania za pomocą metod programowania dynamicznego. Podejście bezpośrednie polega natomiast na poszukiwaniu optymalnej strategii sterowania bezpośrednio na podstawie obserwacji zachowania się sterowanego systemu, bez potrzeby konstruowania modeli. Uczenie się ze wzmocnieniem odpowiada więc bezpośredniemu adaptacyjnemu sterowaniu optymalnemu [19]. Pokazano eksperymentalnie, że takie bezpośrednie podejście oparte na uczeniu się ze wzmocnieniem może niekiedy być wyraźnie bardziej skuteczne niż tradycyjne metody pośrednie [2]. Szerzej zagadnienia zastosowania uczenia się ze wzmocnieniem do sterowania omawiane są np. w pracy [11].

Inną, częściowo pokrewną dziedziną, w której uczenie się ze wzmocnieniem jest stosowane co najmniej równie często, jest robotyka. Celem tych prób jest uzyskanie inteligentnego robota zdolnego do samodzielnego działania w złożonych i na ogół niedeterministycznych środowiskach. Dla takich środowisk sporządzenie tradycyjnie rozumianego oprogramowania sterującego dla robota jest niemożliwe lub bardzo trudne. Często również wykorzystanie metod uczenia się z nadzorem okazuje się niemożliwe ze względu na trudność sformułowania odpowiednich przykładów. Tymczasem zaprojektowanie odpowiedniej funkcji wzmocnienia zazwyczaj nie przedstawia dużej trudności, zwłaszcza, że może to być wzmocnienie opóźnione. System uczący się ze wzmocnieniem, sterujący robotem, obserwuje jako stany środowiska informacje z sensorów robota, a wygenerowane przez niego akcje są używane do aktywowania różnych fizycznych układów robota służących do poruszania się, wykonywania manipulacji. Przeprowadzano liczne eksperymenty z takimi systemami, w zastosowaniu do wykonywania typowych zadań ruchomych robotów, takich jak przechodzenie przez drzwi, nawigowanie w wąskich korytarzach i omijanie przeszkód, znajdowanie drogi, przesuwanie pudełek, ładowanie akumulatorów i inne, zarówno w środowiskach symulowanych, jak i rzeczywistych. Opisy tego typu doświadczeń można znaleźć np. w [12, 14].

Nieco mniej popularny, lecz także obiecujący obszar zastosowań, wiąże się z tworzeniem programów komputerowych grających w gry. Znaczący sukces odniósł wykorzystujący metody uczenia się ze wzmocnieniem program do gry w trik-traka (ang. *backgammon*) [21], zwycięzca komputerowej olimpiady tej gry. Program ten, nazwany *TD-Gammon* i uważany obecnie za jednego z najlepszych graczy na świecie (zarówno wśród ludzi, jak i wśród programów komputerowych), pokonał tradycyjne nie uczące się programy z „ręcznie" zakodowanymi heurystykami uzyskanymi na podstawie wiedzy ekspertów trik-traka, a także wcześniejszy program *Neurogammon* [20] tego samego autora, uczący się z nadzorem z odpowiednio dobranych przykładów. Swoje mistrzostwo osiągnął *TD-Gammon* ucząc się *wyłącznie* na podstawie własnej gry, zaczynając od zerowej wiedzy i nie

korzystając z żadnej pomocy lub innej ingerencji człowieka. Jest to jeden z najbardziej spektakularnych sukcesów nie tylko uczenia się ze wzmocnieniem, lecz także uczenia się maszyn i sztucznej inteligencji w ogóle.

Ostatnio w kilku pracach opisano udane zastosowania uczenia się ze wzmocnieniem do problemów optymalizacji kombinatorycznej, m.in. problemu komiwojażera [10] i problemu szeregowania zadań [24]. Obiecujące są także wyniki uzyskane przy stosowaniu pewnej wersji algorytmu Q-learning do adaptacyjnego znajdowania optymalnych połączeń (routingu) w sieciach z komutacją pakietów [13]. Z pewnością wiele innych obiecujących zastosowań czeka jeszcze na swoich odkrywców.

* * *

Uczeniu się ze wzmocnieniem w jego dzisiejszym kształcie dały początek w pierwszej połowie lat osiemdziesiątych pionierskie prace Andrew G. Barto i Richarda S. Suttona [3, 17], który opracował nie omówiony tu z braku miejsca algorytm AHC (ang. *Adaptive Heuristic Critic*).

Kolejnym krokiem milowym była praca Christophera Watkinsa [22], który odkrył bliskie pokrewieństwo zagadnień badanych wcześniej przez Suttona z teorią stochastycznego programowania dynamicznego i opierając się na tych pokrewieństwach zaproponował algorytm Q-learning. Osiągnięcia te stworzyły podstawę do dalszego rozwoju dziedziny, który wyraźnie nabrał tempa na początku lat dziewięćdziesiątych, aby w ostatnich dwóch-trzech latach zaowocować gwałtownie rosnącą liczbą publikacji, opisujących nowe rezultaty teoretyczne, prace eksperymentalne, zastosowania w coraz bardziej złożonych i realistycznych środowiskach. Świadectwem tego rozwoju są, między innymi, poświęcone uczeniu się ze wzmocnieniem dwa specjalne numery czasopisma *Machine Learning* (pierwszy – Vol. 8, 1992, drugi – ukaże się w tym roku). Na ubiegłorocznej międzynarodowej konferencji na temat uczenia się maszyn (*The Twelfth International Conference on Machine Learning*, Tahoe City, USA) spośród 68 prezentowanych referatów kilkanaście poświęconych było uczeniu się ze wzmocnieniem. Planowane jest wydanie w Stanach Zjednoczonych pierwszej książki w całości poświęconej temu tematowi, autorstwa Barto i Suttona.

Jednocześnie w dziedzinie uczenia się ze wzmocnieniem pozostaje wiele problemów otwartych, które czekają na nowe propozycje rozwiązań. Dotyczą one między innymi szybkości uczenia się, ograniczeń zakresu stosowalności dostępnych algorytmów, integracji uczenia się i planowania, konstruowania systemów o strukturze hierarchicznej. Wszystkie one są właściwie związane z jednym wspólnym celem, którym jest zwiększenie skuteczności istniejących metod na tyle, aby można było je stosować z powodzeniem do dużych i realistycznych problemów.

Oba powyższe spostrzeżenia (gwałtownie zwiększające się zainteresowanie z jednej strony oraz duża liczba wciąż nie rozwiązanych problemów z drugiej) nie pozostawiają wątpliwości, że warto zwrócić uwagę na dziedzinę uczenia się ze wzmocnieniem i usprawiedliwiają popularyzatorską funkcję tego artykułu. Jego ograniczona objętość pozwoliła jedynie na pobieżne omówienie najbardziej podstawowych zagadnień. Czytelników zainteresowanych uzyskaniem bliższych informacji na ten temat autor gorąco zachęca do bezpośredniego kontaktowania się z nim.

LITERATURA

[1] Barto A. G.: Reinforcement learning and adaptive critic methods. D. A. White, D. A. Sofge (Eds.): Handbook of Intelligent Control, pp. 469–491. Van Nostrand Reinhold, New York, 1992

[2] Barto A. G., Singh S. P.: On the computational economics of reinforcement learning. Proceedings of the 1990 Connectionists Models Summer School. Morgan Kaufmann, 1990

[3] Barto A. G., Sutton R. S., Anderson C. W.: Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, and Cybernetics, 13, pp. 835–846, 1983

[4] Bellman R. E.: Dynamic Programming. Princeton University Press, Princeton, NJ, 1957

[5] Bolc L., Zaremba J.: Wprowadzenie do uczenia się maszyn. Akademicka Oficyna Wydawnicza RM, 1992

[6] Carbonell J. G., Michalski R. S., Mitchell T. M.: An overview of machine learning. R. S. Michalski, J. G. Carbonell, T. M. Mitchell (Eds.): Machine Learning: An Artificial Intelligence Approach, Vol. 1. Tioga (obecnie Morgan Kaufmann), 1983

[7] Cichosz P.: Truncating temporal differences: On the efficient implementation of TD(λ) for reinforcement learning. Journal of Artificial Intelligence Research, 2, pp. 287–318, 1995

[8] Cichosz P.: Truncated temporal differences with function approximation: Successful examples using CMAC. Ukaże się w: Proceedings of the Thirteenth European Symposium on Cybernetics and Systems Research (EMCSR-96), 1996

[9] Cichosz P., Mulawka J. J.: Fast and efficient reinforcement learning with truncated temporal differences. Proceedings of the Twelfth International Conference on Machine Learning (ML-95), 1995

[10] Gambardella L. M., Dorigo M.: Ant-Q: A reinforcement learning approach to the traveling salesman problem. Proceedings of the Twelfth International Conference on Machine Learning (ML-95), 1995

[11] Gullapalli V.: Reinforcement Learning and Its Application to Control. PhD thesis, Department of Computer and Information Science, University of Massachusetts, 1992

[12] Long-Ji Lin: Reinforcement Learning for Robots Using Neural Networks. PhD thesis, School of Computer Science, Carnegie-Mellon University, 1993

[13] Littman M. L., Boyan J. A.: A distributed reinforcement learning scheme for network routing. Technical Report CMU-CS-93-165, School of Computer Science, Carnegie-Mellon University, 1993

[14] Mahadevan S., Connell J.: Automatic programming of behavior-based robots using reinforcement learning. Artificial Intelligence, 55, pp. 311–365, 1992

[15] Minsky M. L.: Steps toward artificial intelligence. Proceedings of the Institute of Radio Engineers, 1961. Przedruk w E. A. Feingenbaum, J. Feldman (Eds.): Computers and Thought, McGraw-Hill, New York, 1963

[16] Mitchell T. M.: Machine Learning (w przygotowaniu)

[17] Sutton R. S.: Temporal Credit Assignment in Reinforcement Learning. PhD thesis, Department of Computer and Information Science, University of Massachusetts, 1984

[18] Sutton R. S.: Learning to predict by the methods of temporal differences. Machine Learning, 3, pp. 9–44, 1988

[19] Sutton R. S., Barto A. G., Williams R. J.: Reinforcement learning is direct adaptive optimal control. Proceedings of the American Control Conference. Boston, MA, 1991

[20] Tesauro G.: Neurogammon wins computer olympiad. Neural Computation, 1, pp. 321–323, 1990

[21] Tesauro G.: Practical issues in temporal difference learning. Machine Learning, 8, pp. 257–277, 1992

[22] Watkins C. J. C. H.: Learning from Delayed Rewards. PhD thesis, King's College, Cambridge, 1989

[23] Watkins C. J. C. H., Dayan P.: Technical note: Q-learning. Machine Learning, 8, pp. 279–292, 1992

[24] Zhang W., Dietterich T. G.: A reinforcement learning approach to job-shop scheduling. Proceedings of the Fourteenth Joint International Conference on Artificial Intelligence (IJCAI-95), 1995.

Kontakt z Autorem:
cichosz@ipe.pw.edu.pl
http://www.ipe.pw.edu.pl/~cichosz

Profesjonalny sprzęt w zasięgu ręki

Zainteresowanych zapraszamy na **INFOSYSTEM'96**
do Hali 23, poziom C, stanowisko 311

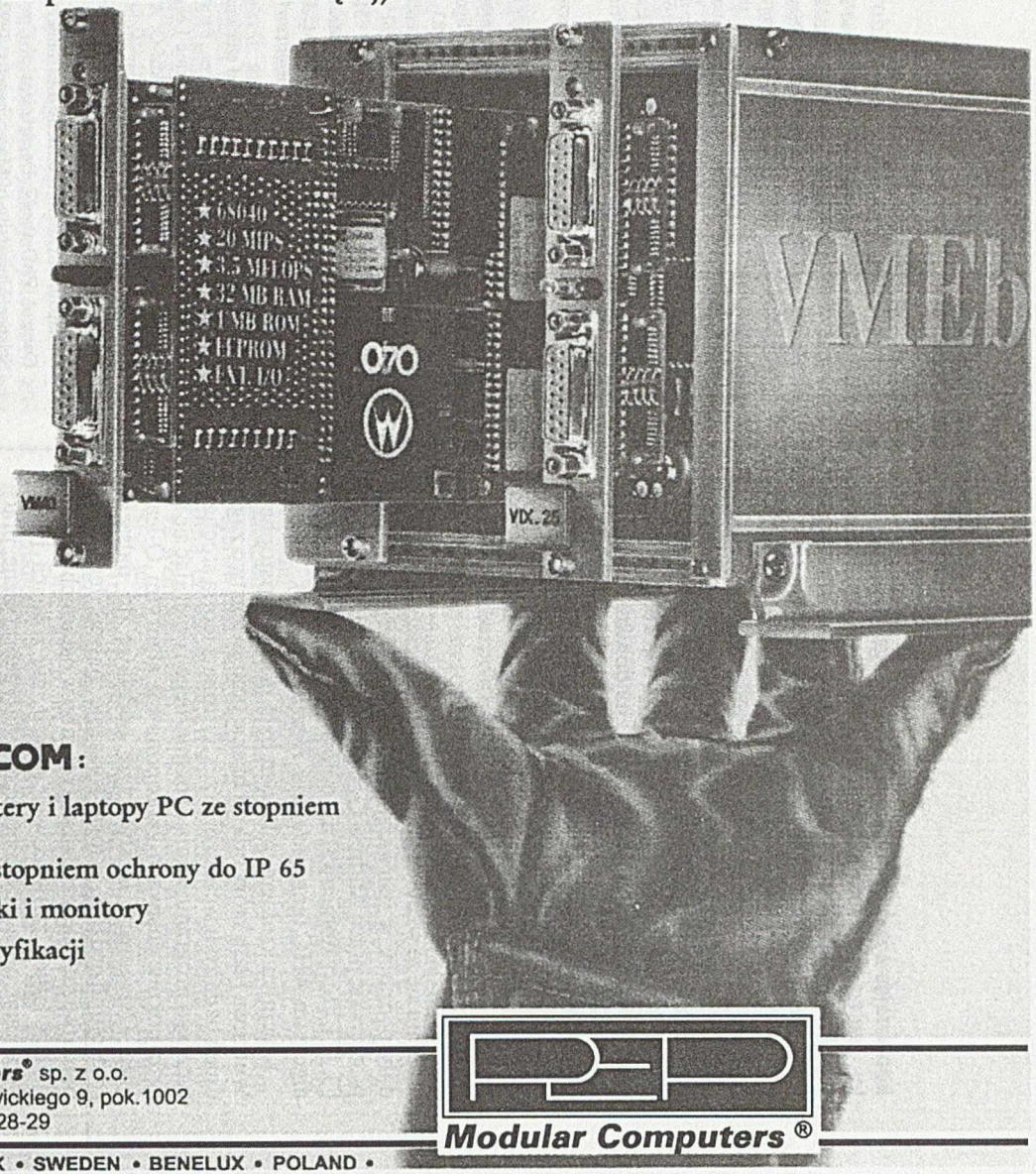
PEP *Modular Computers*® jest największym światowym producentem 3U VME z certyfikatem ISO 9001, specjalizującym się w produkcji sprzętu do pracy w ciężkich warunkach przemysłowych (temperatury pracy do E3: - 55°C ÷ 125°C, udary do 40g, wibracje 5g, EMI, MTBF nawet do 130 lat, zasilanie 12VDC ÷ 220VAC z pełną izolacją galwaniczną, wysoka odporność na przebicie 1500V i więcej).

HARDWARE:

- karty procesorów 1 ÷ 160 MIPS (MC68xxx, PowerPC, Przemysłowe PC)
- **SMART I/O** (real-time mikro PLC) zgodny z Open Modular Architecture Controllers
- karty: wejść/wyjść cyfrowych, analogowo-cyfrowych/cyfrowo-analogowych, wizyjne,
- systemy transmisji: RS-232, RS-422, RS-485, SDLC, HDLC, PROFIBUS, CAN-bus, LONWORKS, BIT-bus, Ethernet, X-25, GPIB, MIL-STD 1553B, AUTOBAHN
- kontrolery: SCSI, SCSI-2, PCMCIA, silników krokowych, sterowników osi

SOFTWARE:

- systemy operacyjne czasu rzeczywistego: OS-9, QNX, VRTX/OS, VxWorks
- IsaGraf for Windows - skrośny system programowania PLC zgodny z IEC 1131-3
- systemy uruchomieniowe
- szkolenia w kraju i za granicą



NOWOŚĆ !!!

Produkty firmy **GERCOM**:

- przemysłowe komputery i laptopy PC ze stopniem ochrony do IP 65
- ręczne terminale ze stopniem ochrony do IP 65
- przemysłowe drukarki i monitory
- zdalne systemy identyfikacji

20 YEARS

CERTIFIED DIN ISO 9001 QUALITY SYSTEM
REG. NO. 0548-01

PEP *Modular Computers*® sp. z o.o.
02-078 Warszawa, ul. Krzywickiego 9, pok.1002
Tel./fax: (22) 25-13-35 / 25-28-29

PEP Modular Computers®

• USA • GERMANY • FRANCE • UK • SWEDEN • BENELUX • POLAND •

01/9/96

Zagadnienia pozyskiwania wiedzy dla sterowników rozmytych (2)

Jarosław Stańczak
Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Tworzenie reguł dla sterowników rozmytych przy użyciu algorytmów statystycznych polega na analizie danych, przedstawiających pożądane sterowania dla określonych obszarów przestrzeni stanów układu. Analiza ta ma na celu wyszukanie obszarów o pewnych cechach wspólnych i przetworzenie ich w odpowiednie reguły dla regulatora (analiza skupień punktów o podobnych właściwościach).

Głównym problemem jest tu dobór optymalnej dla danego problemu liczby reguł, czyli obszarów, na które zostanie podzielona przestrzeń stanów, oraz optymalizacja parametrów funkcji przynależności do zbiorów rozmytych dla wyróżnionych obszarów – tak aby parametry regulacji w największym stopniu odpowiadały oczekiwanym. Wiedza o rozwiązywanym problemie ogranicza się w tym wypadku do dysponowania konkretnymi danymi liczbowymi o obiekcie, czyli do znajomości wartości wyjść z obiektu (zmiennych stanu – o ile to możliwe) i pożądanych wartości sterowań w tych warunkach. Rozwiązanie zaś zadania polega na uogólnieniu tych danych – tak aby w każdym wypadku można było znaleźć właściwe sterowania dla obiektu.

Wykorzystanie statystycznych metod analizy danych do generacji reguł

Opracowano wiele algorytmów, które potrafią automatycznie dokonać podziału przestrzeni stanów i sterowań na obszary, na podstawie których można stworzyć reguły dla sterownika rozmytego. Należą do nich między innymi dwa algorytmy: C-MEANS i ISODATA oraz kilka innych opartych na podobnej zasadzie. Istotą tego typu algorytmów jest taki podział posiadanych danych liczbowych o obiekcie, aby minimalizował on pewne kryterium jakości. Przykładowo dla algorytmu C-MEANS ma ono postać [2]:

$$\{5\} \qquad S(c) = \sum_{k=1}^{k=n}\sum_{i=1}^{i=c}(\mu_{ik})^m(\|x_k - v_i\|^2 - \|v_i - \bar{x}\|^2)$$

gdzie:

n – liczba wektorów danych; c – liczba obszarów; x_k – k-ty wektor danych;

$\bar{x}$ – wartość średnia dla wektorów danych; v_i – środek i-tego obszaru, znalezionego przez algorytm;

m_{ik} – wartość funkcji przynależności k-tego wektora danych do i-tego obszaru;

m – wartość dobierana dla danego wypadku, zwykle dobierana jest z zakresu od 1,5 do 3;

$\|\ldots\|$ – euklidesowa miara odległości.

Kryterium powyższe jest funkcją dostarczonych danych oraz liczby przedziałów, na jakie podzielimy przestrzeń stanów i sterowań. Dlatego też kluczowym problemem jest tutaj minimalizacja kryterium jakości nie tylko przez optymalny podział danych (to zapewnia algorytm), ale także przez odpowiedni dobór wartości c, czyli liczby obszarów, a właściwie reguł sterowania. Niestety, algorytm nie potrafi znaleźć optymalnej wartości tej liczby, wobec czego należy jej szukać metodą prób i błędów, stosując wielokrotnie algorytm dla różnych c. Znalezienie globalnego minimum tego kryterium w zależności od c nie jest zresztą konieczne, gdyż byłoby to pracochłonne i mogłoby prowadzić do zbyt dużej liczby reguł, co nie jest pożądane. Wystarczy zbadać zachowanie wartości kryterium jakości algorytmu dla takich c, które dadzą w efekcie rozsądne liczby reguł. Należy tu wybrać rozwiązanie kompromisowe między dokładnością podziału przestrzeni stanów a szybkością sterownika. Minimalizacja kryterium przez optymalny podział danych sprowadza się do znalezienia takich wektorów v_i, które są środkami skupisk danych – minimalizowana jest wariancja danych wewnątrz każdego wydzielonego obszaru (pierwsza norma euklidesowa we wzorze). Premiowane jest także jak największe oddalenie (zróżnicowanie) wektorów v_i od wektora będącego średnią rozpatrywanych danych $-\bar{x}$ (druga norma w kryterium). Omawiany algorytm podziału danych polega na iteracyjnym przetwarzaniu macierzy $U^{(l)}$, zawierającej wartości funkcji przynależności dla każdego z wektorów danych do poszczególnych klas. Przetwarzanie macierzy trwa dopóty, dopóki zmiany jej współczynników nie będą mniejsze od ustalonej przez użytkownika wartości e.

Algorytm „C-MEANS":

1. Ustalenie liczby c – liczby przedziałów (oczywiście $c > 2$) oraz wartości początkowych macierzy $U^{(l)}$ ($l=1$). Przy nadawaniu wartości początkowych macierzy U, należy pamiętać o warunku:

$$\{6\} \qquad \sum_{i=1}^{i=c}\mu_{ik} = 1$$

oznaczającym, że suma wartości funkcji przynależności k-tej danej do wszystkich klas powinna wynosić 1. Jeśli nie ma innych przesłanek co do wartości $U^{(l)}$, to można ją wypełnić wartościami $1/c$.

2. Obliczenie wektorów v_i – wyrażających środki szukanych obszarów:

$$\{7\} \quad v_i = \sum_{k=1}^{k=n} (\mu_{ik})^m x_k \Big/ \sum_{k=1}^{k=n} (\mu_{ik})$$

dla i = 1, 2...*c*
(oznaczenia jak wyżej).
3. Korekcja współczynników macierzy $U^{(l)}$ (dla $l = l + 1$. Nowe wartości macierzy U oblicza się na podstawie odległości *k*-tego wektora od centrów wydzielonych klas – $d_{ik} = \|x_k - v_i\|$.

a) jeśli *k*-ty wektor danych pokrywa się z centrami (centrum) jakichś wydzielonych obszarów, to:

– jeśli pokrywa się z jednym v_i, *to* $m_{ik} = 1$ dla tej klasy i $m_{ik} = 0$ dla pozostałych;

– jeśli pokrywa się z kilkoma centrami (taki wypadek może wystąpić raczej tylko w początkowych iteracjach), wtedy nadaje im się $m_{ik} < 1$, pamiętając o warunku {6}, dla pozostałych klas $m_{ik} = 0$.

b) jeśli *k*-ty wektor danych nie pokrywa się z żadnym wektorem v_i, wtedy wartości m_{ik} oblicza się według wzoru:

$$\{8\} \quad \mu_{ik} = 1 \Big/ \left[\sum_{j=1}^{j=c} (d_{ik}/d_{jk})^{2/(m-1)} \right]$$

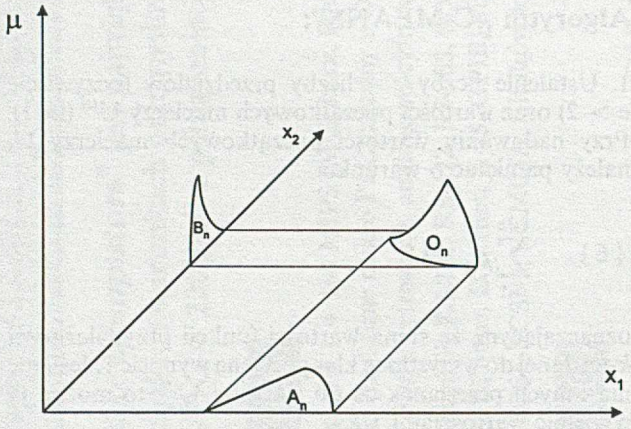
(oznaczenia jak poprzednio).
4. Sprawdzenie kryterium stopu

– jeśli $\|U^{(l-1)} - U^{(l)}\| \leqslant e$ – koniec algorytmu;

– w przeciwnym wypadku przejście do punktu 2.

Algorytm dzieli dane na *c* klas (obszarów); dla potrzeb generacji reguł dla regulatora rozmytego należy ten podział uogólnić na całą przestrzeń możliwych danych. Wymusza to odpowiedni dobór danych do algorytmu, tak aby reprezentowały one dobrze przedziały zmienności zmiennych układu i przez to dobrze określały charakter układu.

Otrzymane w trakcie działania algorytmu wartości m_{ik} określają przynależność danych do pewnych wydzielonych obszarów w przestrzeni zmiennych stanu (wyjściowych) i sterujących. Do generacji reguł potrzebne są funkcje przynależności do zbiorów rozmytych określonych dla konkretnych zmiennych. Jest możliwe otrzymanie takich funkcji przez zrzutowanie otrzymanych wartości funkcji przynależności na konkretną zmienną. Pokazuje to rysunek. Powstałe w ten sposób funkcje przynależności nie mają od razu kształtu trapezu lub trójkąta, lecz można je takim kształtem łatwo aproksymować, stosując metody regresyjne.



Rzutowanie wartości funkcji przynależności do obszaru O_n na zmienne układu

Przedstawiony powyżej sposób tworzenia reguł na podstawie statystycznej analizy danych o układzie regulacji jest dość charakterystyczny dla tej klasy algorytmów. Jak widać, jest on nieco skomplikowany obliczeniowo.

Wykorzystanie sztucznych sieci neuronowych w procesie tworzenia reguł

Szybki rozwój teorii i zastosowań sztucznych sieci neuronowych dał nowe możliwości tworzenia i wykorzystania układów opartych na logice rozmytej. Wyrażają się one zarówno w zastosowaniu urządzeń o nowej architekturze, łączącej cechy sieci neuronowych i układów rozmytych, jak i nowych możliwościach poszukiwania reguł dla urządzeń działających na zasadzie wnioskowania rozmytego.

Sieci neuropodobne [3-6] są systemami, których inspiracją jest układ nerwowy organizmów zwierzęcych i sposób, w jaki jest w nim przetwarzana informacja. Przetwarzanie informacji w sieci dokonuje się dzięki odpowiednio połączonym sztucznym neuronom. Sieci takie, podobnie jak ich biologiczne pierwowzory, mają zdolność uczenia się. Uczenie to polega na utworzeniu odpowiednich połączeń neuronów, gdzie ważnym parametrem jest siła połączeń między poszczególnymi neuronami. Połączenia te charakteryzowane są za pomocą współczynników zwanych *wagami połączeniowymi*. Wyuczone wartości wag połączeniowych odpowiednio modyfikują przechodzące przez neuron sygnały. Wyjście z *k*-tego neuronu sieci jest obliczane ze wzoru:

$$\{9\} \quad y_k = f\left(\sum_{i=1}^{i=n} u_i w_{ik} - \Theta_k \right) = f(U^T W_k - \Theta_k)$$

gdzie:

y_k – wyjście neuronu; f(..) – funkcja aktywacji neuronu, najczęściej sgn(*x*) lub $1/(1 + \exp(-x))$;

u_i – *i*-te wejście neuronu; *U* – wektor wejść do neuronu;

w_{ik} – waga połączeniowa przyporządkowana *i*-temu wejściu;

W_k – wektor wag połączeniowych neuronu *k*; Q_k – próg aktywacji neuronu.

Opracowano wiele różnych modeli sieci neuropodobnych, znacznie różniących się topologią, funkcjami, możliwościami, a także sposobami ich uczenia. Niektóre z nich mogą być dużą pomocą przy tworzeniu reguł dla układów rozmytych. Podobnie jak w wypadku algorytmów statystycznych, sieci neuronowe można wykorzystać do analizy danych i tworzenia na ich podstawie reguł dla „rozmytych" systemów ekspertowych. Sposób działania jest tu też podobny – należy podzielić dane na klasy, z których powstaną reguły. Każda klasa będzie zawierać dane o zbliżonych cechach. O optymalnej liczbie klas (obszarów), na które zostanie podzielona przestrzeń zmiennych na podstawie posiadanych danych, jak i o ich przynależności, zadecyduje sama sieć. Do rozwiązania tak postawionego zadania najlepiej nadają się tzw. *sieci samoucz ące się*, czyli tzw. *sieci trenowane bez sygnału uczącego*. Najczęściej wykorzystywane są tu dwa rodzaje sieci: ART i tzw. *sieć Kohonena* (patrz np. [4] i [6]).

W praktycznych realizacjach należy raczej mówić o algorytmach uczenia opartych na tzw. *rezonansie adaptacyjnym* (ART) i uczeniu według schematu podanego przez Kohonena.

Zjawisko samouczenia się polega na tworzeniu klas wzorców, na podstawie prezentowanych sieci danych. Proces dzielenia danych na klasy zachodzi samoczynnie, bez nadzoru człowieka i oparty jest na właściwościach sieci użytej do rozwiązania problemu. W sieciach samouczących się najczęściej stosowany jest *algorytm uczenia konkurencyjnego*. Jest on oparty na zasadzie zwycięzca bierze wszystko, oznaczającej, że prezentowany sieci przykład może być zaklasyfikowany tylko do jednej grupy danych, najbliższej prezentowanemu przykładowi. Nawet jeśli jest to pierwszy przykład pokazywany sieci w procesie uczenia, to zawsze któryś neuron ma wartości wag połączeniowych najbliższe tej danej (*neuron zwycięzca*). W procesie uczenia wagi połączeniowe tego neuronu zostają skorygowane tak, aby były jeszcze bliższe tej danej.

W niektórych schematach uczenia sieci (np. w sieci Kohonena) korygowane są też wagi neuronów sąsiadujących ze zwycięzcą. O ile prezentowany przykład będzie zbyt odległy od wszystkich poznanych wcześniej wzorców, to sieć tworzy nową klasę danych. Wynika z tego, że generowane w trakcie nauki sieci klasy są rozłączne. Sieć nie wygeneruje nam wobec tego wartości funkcji przynależności danej do danego obszaru lub też kilku sąsiednich obszarów, gdyż jej wyjście określi jedynie fakt przynależności lub nie przynależności do tej klasy. Może to być jednakże dobry punkt wyjścia do generacji funkcji przynależności do wyodrębnionego obszaru np. na podstawie odległości danej od wektora wyrażającego środek danej klasy[1]. Linie podziału między klasami będą wtedy interpretowane nie jako sztywne granice, a jako miejsca, w których zrównują się wartości funkcji przynależności do różnych obszarów.

Algorytm podziału danych dla sieci Kohonena:

1. Ustalenie liczb: c – liczby neuronów, N_0 – wektora indeksów neuronów „sąsiednich" do zwycięzcy, t_{max} – maksymalnej liczby iteracji, e – minimalnej zmiany wag połączeniowych z iteracji na iterację ($e>0$), $m_{0,i}$ – startowej wartości współczynnika wygaszania procesu uczenia (współczynnik ten ma za zadanie zmniejszanie zmian wag połączeniowych wraz z numerem iteracji, $m_0>1$), $a_{i,k,0}$ – współczynnika uczenia ($a_{i,k,0} \in (0,1)$), oraz startowego wektora wag połączeniowych $w_0 = (w_{1,0}, w_{2,0}, w_{3,0}..., w_{c,0})$. Należy tu pamiętać, aby startowe wartości wag połączeniowych generować losowo, unikając ich symetrii, najlepiej jako małe liczby z przedziału (0,5; 0,5).
2. Dla $t = 1... t_{max}$ (kolejne iteracje) :
3. Dla $k = 1...n$ (kolejne dane – macierz danych $X = \{x_1, x_2,..., x_n\}$) :

a) obliczenie euklidesowej odległości między bieżącą daną (x_k) a wektorami wag połączeniowych sieci ($w_{i,t-1}$) z poprzedniej iteracji, dla $i=1...c$:

$$d_{ik}^2 = \left\| x_k - w_{i,t-1} \right\|^2;$$

b) znalezienie zwycięzcy (najmniejszej odległości) oraz kolejnych rosnących odległości;

c) korekcja wektora wag połączeniowych neuronu zwycięskiego (oznaczmy go jako l):

$$w_{l\,t} = w_{l\,t-1} + a_{l\,t}(x_k - w_{l\,t-1})$$

d) Korekcja wektorów wag połączeniowych neuronów sąsiednich, opisanych wektorem N_t:

$$w_{i,t} = w_{i,t-1} + a_{i,t}(x_k - w_{i,t-1})$$

e) $k=k+1$; przejście do punktu a).

4. Obliczenie wskaźnika zmian wag połączeniowych w kolejnych iteracjach:

$$E_t = \sum_{i=1}^{i=c} \left\| w_{i,t} - w_{i,t-1} \right\|^2$$

5. Jeśli $E_t \leqslant e$ – koniec algorytmu, w przeciwnym wypadku:
   – korekcja współczynników nauki, np.:
   $a_{i,t} = m_{0,i} - t(m_{0,i} - 1)/t_{max}$;
   – ewentualna korekcja wektora sąsiedztwa N_t;
   – $t = t+1$; przejście do punktu 3.

Istnieje również możliwość modyfikacji sieci neuronowej tak, aby zwycięzców było kilku. Będą to neurony reprezentujące klasy najbardziej zbliżone do przykładu prezentowanego sieci. Wartości funkcji przynależności można określić na podstawie wartości wyjść ze zwycięskich klas biorąc pod uwagę wartości tych wyjść oraz nie zapominając o warunku (6).

Znane są różne algorytmy uczenia się sieci – tak zmodyfikowane, aby oprócz podziału danych na grupy podawały one również wartość funkcji przynależności do tej klasy. Przykładem może tu być zmodyfikowany algorytm Kohonena, obliczający wartości funkcji przynależności dla klasyfikowanych danych, zamieszczony w pracy [1]. Ciekawostką jest fakt, że sposób działania tego algorytmu jest bardzo zbliżony do opisanego wcześniej algorytmu C-MEANS, z tym, że zamiast ustalonej liczby obszarów jest tu ustalona liczba neuronów, z których każdy może stanowić oddzielną klasę danych.

Oprócz sieci bez nadzoru, wykorzystuje się również sieci *trenowane z sygnałem uczącym*. Szczególnie przydatna może tu być sieć typu perceptronowego z wsteczną propagacją błędu (ang. *backpropagation* lub BP). Szersze informacje o tego typu sieci można znaleźć np. w pracach [3-6]. Można ją wykorzystywać do generacji funkcji przynależności dla zbiorów rozmytych, na podstawie posiadanych już przykładów danych wraz z określonymi dla nich wartościami funkcji przynależności m. Może to być dobre uzupełnienie dla jakiegoś z algorytmów klasyfikacji danych. Sieć ta ma właściwość interpolacji przebiegu funkcji na podstawie dostarczonych jej danych. Po nauczeniu sieci funkcji przynależności dla wszystkich zmiennych może ona służyć jako część fuzyfikacyjna sterownika rozmytego. Ma to tę dodatkową zaletę, że otrzymane w ten sposób funkcje przynależności nie będą miały odgórnie narzuconej postaci (trójkątnej lub trapezoidalnej), lecz taką, która będzie najlepiej pasować do podanych danych (a przez to i do badanego zagadnienia). Przykład takiego rozwiązania jest przedstawiony w pracy [5].

* * *

Właściwy dobór parametrów regulatora sterującego danym procesem jest kluczem do otrzymania dobrego układu regulacji. Proces projektowania zestawu reguł dla

[1] Wektor wyrażający środek danej klasy jest pamiętany dla każdego neuronu wyjściowego sieci w postaci jego wag połączeniowych.

regulatora rozmytego jest procesem szczególnie trudnym i pracochłonnym. Praca niniejsza jest próbą przedstawienia problemów występujących w tym procesie oraz sposobów ich przezwyciężania. Zawiera ona opisy trzech metod pozyskiwania zestawów reguł dla regulatorów: wykorzystując wiedzę eksperta w danej dziedzinie, przez statystyczną analizę danych liczbowych o obiekcie oraz wykorzystując do analizy danych sieci neuronowe. Ta ostatnia metoda tworzenia reguł wydaje się obecnie najefektywniejsza – zwłaszcza, że istnieją bardzo obiecujące próby połączenia zalet przetwarzania informacji przy pomocy układów bazujących na logice rozmytej i sieci neuronowych. Idea zestawienia w jeden układ sieci neuronowej i systemu opartego na wnioskowaniu rozmytym wydaje się obecnie jednym z najbardziej przyszłościowych podejść w dziedzinie sztucznej inteligencji. Umożliwia ona połączenie zdolności do nauki charakterystycznej dla sieci neuronowych z możliwością przedstawienia tej wiedzy w postaci prostych, łatwych do zrozumienia i interpretacji reguł. Pozyskiwanie reguł działania dla takiego układu nie będzie stanowiło już problemu, gdyż mechanizm taki będzie weń wbudowany.

LITERATURA

[1] Bezdek J. C.: Computing with Uncertainity. IEEE Communications Magazine, September 1992

[2] Czogała E., Pedrycz W.: Elementy i metody teorii zbiorów rozmytych. PWN, Warszawa 1985

[3] Hecht-Nielsen R.: Neurocomputing. Addison-Wesley Publishing Company, Inc, 1990

[4] Hertz J., Krogh A., Palmer R. G.: Wstęp do teorii obliczeń neuronowych. WNT Warszawa 1993

[5] Hoyashi I., Nomura H., Wakami N.: Artificial Neural Network-Driven Fuzzy Control and Its Application to the Learning of Inverted Pendulum System. IEEE Tokyo Section, Denshi Tokyo No. 29, 1990

[6] Korbicz J., Obuchowicz A., Uciński D.: Sztuczne sieci neuronowe – podstawy i zastosowania. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1994

[7] Sugeno M., Yasukawa T.: Fuzzy-Logic-Based Approach to Qualitative Modeling. IEEE Transactions on Fuzzy Systems, Vol. 1, February 1993.

Zastosowanie komputera...

dokończenie ze s. 3

Na rysunkach 5, 6 i 7 są pokazane fragmenty danych wyjściowych systemu, uzyskane w warunkach Instytutu Informatyki Uniwersytetu Jagiellońskiego (około 50 prowadzących zajęcia, 9 sal, około 300 studentów).

Godzina	Poniedziałek	Wtorek	Środa	Czwartek	Piątek
8 - 9				SK gr.1 s 215	
9 - 10				SK gr.1 s 215	
10 - 11				SK w s 215	
11 - 12				SK w s 215	
12 - 13			UC gr.1 s 221		
13 - 14			UC gr.1 s 221		
14 - 15			UC w s 221		
15 - 16			UC w s 221		
16 - 17			OP w s 215		
17 - 18			OP w s 215		
18 - 19					
19 - 20					

Rys. 7. Fragment przykładowego rozkładu dla prowadzących zajęcia dr M. Skomorowski

Przedstawiony tu komputerowy system automatycznego układania rozkładu zajęć został zrealizowany w języku C w systemie operacyjnym MS DOS ([3]). Tekst źródłowy zawiera około trzech tysięcy wierszy. Podstawowymi strukturami danych w systemie są pliki na dysku twardym. Zatem wielkość danych wejściowych jest ograniczona pojemnością dysku twardego.

LITERATURA

[1] Even S., Itai A., Shamir A.: On the Complexity of Timetable and Multicommodity Flow Problems, SIAM Journal on Computation, 5, 1975, 691–703

[2] Garrey M.R., Johnson D.S., Stockmeyer L.: Some Simplified NP-complete Graph Problems. Theoretical Computer Science 1, 1974, 237–267

[3] Jendrzejczyk B.: Automatyczne układanie rozkładu zajęć dla wyższej uczelni – projekt i implementacja (praca magisterska), Instytut Informatyki UJ, Kraków 1994

[4] Lembas J., Skomorowski M., Woźniak R.: Pewien algorytm układania rozkładu zajęć w szkole wyższej, Zeszyty Naukowe UJ, Prace Informatyczne, Zeszyt 2, 1985, 19–28

[5] Schmidt G., Strohlein T.: Timetable construction – an annotated bibliography, The Computer Journal, vol.23, 1980, 307–316

[6] Skomorowski M., Woźniak R.: Komputerowy system układania rozkładu zajęć dla uczelni wyższej, Informatyka, 1985, 7, 22–23.

Książki nadesłane Wydawnictwa Naukowo-Techniczne

Maurice J. Bach: Budowa systemu operacyjnego UNIX. Z ang. przełożyły: Mirosława Miłkowska i Janina Mincer-Daszkiewicz. WNT, Warszawa 1995 r., wyd. 1, s. 506, cena 22,0 zł
ISBN 83-204-1779-1

W książce przedstawiono powszechnie dziś stosowany system operacyjny Unix. Omówiono zagadnienia istotne z punktu widzenia projektanta systemów, takie jak algorytmy szeregowania procesów, strategie zarządzania pamięcią operacyjną i dyskową. Dokładnie opisano jądro systemu Unix oraz wszystkie stosowane w tym systemie algorytmy zarządzania zasobami. Podano także wartości parametrów systemowych, których nie może zmienić nawet administrator, a które w istotny sposób wpływają na działanie systemu.

Książka pomaga zrozumieć, jak działa jądro systemu, a także jak współdziałają i współzawodniczą programy pisane przez użytkowników. Zamieszczono w niej wiele przykładów, a każdy rozdział kończy się zestawem ćwiczeń.

Książka jest przeznaczona zarówno dla projektantów systemów operacyjnych, jak i dla zaawansowanych użytkowników systemu Unix oraz studentów kierunków informatycznych. **(t)**

Ewolucja metod zarządzania pamięcią w dużych systemach komputerowych

Josch Krischer
Comparex Informationssysteme GmbH

Przeglądając wstecz architekturę każdej generacji dużych systemów komputerowych IBM ten rozdział wyjaśni, jak kolejne rodziny procesorów uzyskiwały bardziej wydajny dostęp do danych. W rozdziale tym znajdą się pewne sugestie dotyczące doboru pojemności pamięci.

Pojemność oraz czas dostępu dla nośników danych zawsze odgrywały kluczową rolę w ogólnej ocenie wydajności oraz przepustowości komputerów a największą różnicę pomiędzy dużymi systemami komputerowymi a innymi platformami komputerowymi nie były MIPS-y lecz wielkość wewnętrznej oraz zewnętrznej pamięci oraz możliwości przenoszenia ogromnej ilości danych w sposób możliwie efektywny.

Powstające nowe programy, np. obróbka obrazu, przetwarzanie obiektowe, multimedia, rozpoznawanie mowy, programy samouczące się oraz inne wymuszają powstawanie nowych nośników danych charakteryzujących się większymi rozmiarami oraz prędkością transmisji. Transmisja danych może być tu rozumiana jako transmisja wewnętrzna - angażująca procesory, pamięć centralną oraz rozszerzoną - lub transmisja zewnętrzna pomiędzy pamięcią centralną oraz pomocniczą.

Wielkość oraz sposób wykorzystania pamięci wirtualnej oraz rzeczywistej stanowią główne różnice pomiędzy różnymi generacjami dużych systemów komputerowych firmy IBM. Rozdział ten stanowi przegląd kolejnych generacji.

System/360

System/360 był pierwszym standardem architektury dużych systemów komputerowych - nową koncepcją zapewniającą przenośność systemu operacyjnego oraz oprogramowania. W ramach tej architektury podstawę adresowania pamięci głównej stanowił bajt. Przy 24 bitowym adresowaniu dawało to pojemność w wysokości 16MB. Niemniej jednak wielkość ta była czysto akademicka, gdyż pamięć robiona wówczas z rdzeni magnetycznych była nieporęczna i niezwykle kosztowna. W praktyce maksymalna dostępna pamięć w komputerach IBM 360/65 wynosiła 2 MB, a spora liczba maszyn wyposażona była w zaledwie pół megabajta.

Na początku lat siedemdziesiątych pojawiły się pierwsze układy pamięci oparte o układy półprzewodnikowe 1 KB.

Systemem operacyjnym dla dużych użytkowników był OS/360 MFT (wielozadaniowy ze stałą liczbą zadań). Mechanizm ochrony oparty był na 4 bitowych kluczach przyporządkowanych dla każdego 2 KB segmentu pamięci i ograniczał liczbę zadań do 16. SCP, Reader oraz Writer wykorzystywały na stałe trzy klucze, zatem tylko 13 regionów mogło być wykorzystanych przez programy.

Bardziej efektywne wykorzystanie pamięci zapoczątkował system MVT (wielozadaniowy ze zmienną liczbą zadań), który w zasadzie był podobny do systemu MFT poza tym, że nie występowały w nim stałe partycje (zob. rys.1 na końcu artykułu). Pamięć dla kolejnych zadań była przydzielana dynamicznie tak, jak to było zdefiniowane przez parametr REGION w JCL lub procedurach czytających. Dwa podstawowe ograniczenia to oczywiście wielkość pamięci oraz liczba użytkowników. Wymusiły one wprowadzenie mechanizmu pamięci wirtualnej.

System/370

W Systemie/360 żaden program nie mógł być uruchomiony, dopóki nie został całkowicie załadowany do pamięci, niezależnie od faktu, że w całym aktywnym regionie tylko mały jego fragment był bezpośrednio używany. Koncepcja pamięci wirtualnej nie była nowa - ten pomysł wykorzystano w 1961 roku w komputerze ATLAS - wspólnym przedsięwzięciu Uniwersytetu w Manchester (Wielka Brytania) oraz Ferranti. Ta koncepcja pozwala przechowywać większą część uruchamianego kodu w pamięci pomocniczej podczas, gdy tylko aktywna część kodu i ostatnio używane adresy pozostają w pamięci głównej. Każdy użytkownik ma dostęp tylko do dużych rozmiarów pamięci wirtualnej; sprzęt oraz oprogramowanie dba o to, by odpowiednie instrukcje oraz ich operandy były obecne w pamięci głównej w chwili, gdy jest to potrzebne.

W architekturze komputerów IBM, translacja pomiędzy adresami wirtualnymi (widzianymi przez programistę) oraz rzeczywistymi (widzianymi przez procesor) odbywa się dzięki sprzętowym rozwiązaniom zwanym Dynamiczną Translacją Adresów (DAT). Przestrzeń adresów wirtualnych jest podzielona na segmenty oraz strony. DAT wykorzystuje tablicę wbudowaną i obsługiwaną przez segment SCP oraz tablice stron. Rejestr zwany Rejestrem Kontrolnym 1 zawiera Segment Table Origin (STO). STO + przesunięcie segmentu (część adresu wirtualnego) wskazuje na Page Table Origin (PTO). PTO + przesunięcie strony wskazuje na na rzeczywisty adres w Page Table Entry (PTE), a PTE wraz z przesunięciem bajtowym jest adresem rzeczywistym (zob. rys. 2).

Jedno z pól w PTE jest flagą zwaną Page Invalid Flag. Wartość „1" w tym polu oznacza, że żądana strona nie jest obecna w pamięci rzeczywistej. W takim przypadku program wywołuje procedurę weryfikacyjną (Page Translation Exception lub „brak strony"). Żądana strona musi zostać odnaleziona w pamięci pomocniczej oraz skopiowana do pamięci operacyjnej zanim program wznowi wykonywanie kolejnych instrukcji. Ten proces nazywany jest stronicowaniem. Nowa strona jest przechowywana w pamięci głównej w miejscu wskazywanym przez algorytm „Least Recently

Used" (LRU), według którego usuwana jest z pamięci strona najdłużej nieużywana. Do pamięci głównej zostają dodane dwa nowe bity, Reference (R-bit) oraz Change (C-bit zwany także bitem naruszalności). R-bit jest wykorzystywany przez algorytm LRU, natomiast C-bit zostanie ustawiony na „1", jeśli bieżąca strona została zmodyfikowana. Jeśli algorytm LRU postanowi użyć ramki pamięci głównej zawierającej stronę z ustawionym C-bit, wówczas taka strona musi zostać skopiowana do pamięci pomocniczej zanim ten obszar pamięci głównej zostanie ponownie wykorzystany.

IBM zaimplementowała pamięć wirtualną w systemie operacyjnym OS/VS1, a następnie w OS/VS2 Release 1. Wielkość pamięci wirtualnej wynosiła 16 MB i była dzielona dla wszystkich użytkowników. Dlatego ten system operacyjny znany był jako Single Virtual Storage (SVS). Rozmiar strony został zwiększony do 4 KB, lecz ponieważ ochrona zadań była wykonywana w oparciu o klucz ochrony pamięci, liczba równoległych zadań była nadal ograniczona.

Minęły raptem dwa lata od pojawienia się DAT, gdy zaprojektowano system operacyjny, w którym niezwykle efektywnie wykorzystano koncepcję pamięci wirtualnej: OS/VS Version 2 Release 2 czyli MVS. System MVS wprowadził pojęcie Wielokrotnych Przestrzeni Adresowych, każda o rozmiarze 16 MB oraz każda z nich zawierająca: obszar wspólny, obszar systemowy oraz obszar prywatny. Docelowym rozmiarem obszaru prywatnego było 8-10 MB. Użytkownicy zostali uwolnieni od limitu równoległych zadań, gdyż ochrona pamięci odbywała się dzięki mechanizmom translacji (zob. rys. 3.)

Pionierzy systemu MVS byli zaskoczeni, ponieważ system SVS pracował efektywnie na procesorach z 1 MB pamięci operacyjnej lub nawet mniej, to MVS potrzebował 2 MB do samego uruchomienia oraz co najmniej 4 MB dla uzyskania akceptowalnej wydajności. Maksymalny rozmiar pamięci operacyjnej na maszynach 370/158 wynosił 4 MB, a na maszynach 370/168 wynosił 8 MB. Większość dużych systemów komputerowych była wyposażona w mniejszą ilość pamięci niż maksymalna, co powodowało niezadowolenie pierwszych użytkowników MVS. MVS miał większe wymagania pamięciowe niezbędne do przechowywania segmentów oraz tablic stron. Niewystarczająca pamięć rzeczywista powodowała nadmierne stronicowanie oraz znaczne obniżenie wydajności systemu. Była to sytuacja krytyczna dla pierwszych wersji MVS, gdyż procedury wymiany stron mogły zdominować dostęp do urządzenia stronicującego kosztem właściwych aplikacji. Zostało to ulepszone wraz z wprowadzeniem przełączanych zbiorów danych oraz szybszych urządzeń stronicujących takich jak „bębny", dyski półprzewodnikowe oraz kontrolery DASD dysponujące pamięcią cache.

Problemem jest optymalny dobór wielkości pamięci rzeczywistej. MVS buduje tablicę stron Page FrameTable (PFT) a każda ramka pamięci operacyjnej ma swoją pozycję w tej tablicy. Pole licznika (Unreferenced Interval Count –UIC) każdej takiej pozycji śledzi wiek strony w pamięci operacyjnej. Pole to zawiera liczbę z przedziału od 0 do 255 i reprezentuje czas w sekundach, jaki upłynął od ostatniego odwołania do tej strony – wiek strony. Wartość w tym polu jest ustawiana przez procedurę uaktualniającą licznik UIC, która każdorazowo sprawdza wartość wspomnianego wyżej R-bitu. Średnia licznika UIC (AUIC) jest przechowywana w raporcie RMF. Wartość mniejsza od 30 zwykle oznacza, że pamięć rzeczywista jest zbyt mała; mniejsza niż 10 oznacza wysoką aktywność stronicowania pamięci.

Innym napotkanym, szczególnie przez dużych użytkowników, problemem było ścieśnianie rozmiarów obszaru prywatnego. Każda powstająca w IBM nowa wersja programu lub składnika MVS, jak np. VTAM, wykorzystywała więcej pamięci dla obszaru wspólnego (Common Service Area), redukując wielkość obszaru prywatnego oraz prowadząc do ograniczeń pamięci wirtualnej. Chwilowe załagodzenie tego problemu przyszło razem z pojawieniem się nowego rozwiązania sprzętowego: Podwójnej Przestrzeni Adresowej (Dual Address Space) oraz nowego składnika systemu MVS, korzystającego z niego - Usługi Pamięci Skrośnej. Oba te rozwiązania ustanowiły nowy sposób przenoszenia danych pomiędzy dwiema przestrzeniami adresowymi. Rejestr Kontrolny 7 zawierał początek tablicy segmentów dla drugiej przestrzeni adresowej.

Jeśli chodzi o pamięć rzeczywistą, to sytuacja wygląda nieco lepiej. MVS/SP173 wykorzystuje dwa bity (Rozszerzone Adresowanie - EA) na każdej pozycji tablicy stron w celu rozszerzenia rzeczywistych adresów tak, by można było zaadresować do 64 MB pamięci rzeczywistej. Niemniej jednak, możliwość wykorzystania pamięci operacyjnej powyżej 16 MB była ograniczona, ponieważ rozmiar pamięci wirtualnej był limitowany do 16 MB przez format instrukcji; rozmiar prywatnego obszaru w niektórych przypadkach ograniczony do 4 MB lub mniej i jedynym rozwiązaniem była zmiana architektury.

System/370 Extended Architecture

System/370 Extended Architecture zrewolucjonizował dziedzinę operacji I/O wprowadzając podsystemy dynamicznych kanałów (Dynamic Channel Subsystem) oraz 31-bitowego adresowania dla pamięci rzeczywistej jak i wirtualnej, co rozszerza zakres przestrzeni adresowej do 2 GB. Korzyść była natychmiast odczuwalna nawet bez zmiany oprogramowania, gdyż niektóre składniki MVS (np. SQA oraz CSA) mogły być przesunięte powyżej obszaru 16 MB, zezwalając na powiększenie obszaru prywatnego.

Aczkolwiek architektura zezwalała na użycie do 2 GB pamięci rzeczywistej, w komputerach było na ogół znacznie mniej. Użytkownik IBM seria 308x mógł użyć maksimum do 32 MB, a pierwsze maszyny serii 3090 obsługiwały nie więcej niż 64 MB na procesor. Użytkownicy PCM byli w lepszym położeniu; mogli użyć do 64 MB na procesorze Comparex 7/88 MPs lub NAS AS/9000 oraz 256 MB na 7/9X lub XL odpowiednio.

Seria 3090 spowodowała pojawienie się pamięci rozszerzonej. W tym samym czasie pamięć główna zmieniła swą nazwę na pamięć centralną. Pamięć rozszerzona adresowana jest blokami 4 KB, podczas gdy w pamięci centralnej adresowany jest każdy bajt. 32 bitowy adres pamięci rozszerzonej gwarantuje przestrzeń adresową do 16 TB (terabajtów), co w praktyce oznacza przestrzeń zupełnie nieograniczoną. W pamięci rozszerzonej nie ma bitu odwołania, zmiany, lub ochrony pamięci ale i nie ma do niej bezpośredniego dostępu z podsystemów I/O lub z procesora. Gdy procesor żąda danych, które znajdują się w pamięci rozszerzonej, cały blok 4 KB jest przenoszony do pamięci centralnej. Ten proces jest inicjowany przez system operacyjny instrukcją „PAGEIN". Operacja ta jest synchroniczna - tzn. procesory podłączone do tego samego systemu kontroli (System Control Element) nie będą wykonywały instrukcji aż do zakończenia przenoszenia bloku danych. W praktyce każdy procesor może wykonać instrukcję o ile jej kod oraz operandy znajdują się w buforze dużej szybkości (High Speed Buffer), lecz w większości przypadków ta aktywność jest mało znacząca. W podobny sposób wykonywana jest instrukcja „PAGEOUT", by zainicjować przeniesienie bloku z pamięci centralnej do rozszerzonej. Czas trwania takiej operacji w obie strony wynosi około 75 mikrosekund dla procesorów 3090 oraz 3090E, z czego jedna trzecia to operacje sprzętowe a dwie trzecie to programowe. Czas ten zo-

stał nieznacznie zredukowany do 72 mikrosekund w modelu S oraz do 70 mikrosekund w modelu J oraz 9021 H0 dzięki szybszemu dostępowi do pamięci oraz szybszym procesorom.

Wprowadzenie pamięci rozszerzonej było wielką niespodzianką, gdyż do momentu ogłoszenia Enterprise System Architecture/370 (ESA/370) jedynym jej zastosowaniem było stronicowanie i przełączanie. Prawdą jest, że stronicowanie przy użyciu pamięci rozszerzonej jest szybsze, a wiarygodność danych w takiej pamięci jest wyższa dzięki podwójnemu bitowi korekcji błędu (chociaż rozszerzona korekcja mogłaby być także zaimplementowana w pamięci centralnej, tak jak miało to miejsce w komputerach typu mainframe firmy Hitachi). Kolejnym argumentem przemawiającym za relatywnie małym rozmiarem pamięci centralnej było to, że czas wykonywania Licznik UIC, który jest proporcjonalny do wielkości pamięci centralnej nie był rozgraniczalny z następujących przyczyn:

- Efekt ten jest istotny przy wykorzystaniu więcej niż 64 MB pamięci.
- Każdy synchroniczny transfer bloku zatrzymuje przyłączone procesory na czas równoważny wykonaniu około 1500 instrukcji.
- W każdym kolejnym modelu maksymalna pamięć centralna była podwajana.

Najbardziej przekonywującym argumentem za użyciem pamięci rozszerzonej do stronicowania był fakt, że można było stosować wolniejsze układy pamięci, które były tańsze od tych, stosowanych w pamięci centralnej.

Osobiście wierzę, że powody zastosowania pamięci rozszerzonej w podstawowych modelach 3090 były odmienne. Średni czas wykonania instrukcji jest sumą czasu działania procesora (gdzie instrukcje oraz operandy są pobierane tylko z HSB) oraz średnich dodatkowych operacji w pamięci niezbędnych do przeniesienia instrukcji oraz ich operandów do HSB. Te nakłady są odwrotnie proporcjonalne do współczynnika trafień (hit-ratio) w HSB oraz czasu dostępu do pamięci i powinny stanowić około 20–25% średniego czasu wykonania instrukcji procesora. Pierwsze modele 3090 wykorzystywały zaledwie 64 KB HSB (ograniczona przez osiągalną gęstość upakowania układów bipolarnych oraz przestrzeń upakowania TSM). Pamięć większa niż 64 MB mogła powodować gorszy współczynnik trafień oraz wyższe nakłady. Takie podejście stosowano także w następnych modelach, w których maksymalna dostępna pamięć centralna była podwojona – w modelach E z szybszymi układami pamięci centralnej oraz modelach S z 128 KB HSB.

Enterprise System Architecture/370

Rzeczywiste zastosowanie pamięci rozszerzonej rozpoczęło się razem z wprowadzeniem na rynek architektury ESA/370. Podczas, gdy architektura XA była pionowym rozszerzeniem pamięci wirtualnej – przestrzeń adresowa rosła z 16 MB do 2 GB – architektura ESA stanowiła poziome rozszerzenie poprzez dodanie dodatkowej (do 2 GB) przestrzeni danych, w której jednostką adresowaną był bajt, zawierający jedynie dane, a nie wykonywalny kod (zob. rys. 5) Strony przestrzeni danych mogą znajdować się w pamięci centralnej, rozszerzonej lub pomocniczej. Równie dobrze, jak większa pamięć wirtualna, przestrzenie danych zapewniają lepszą ochronę danych izolując je od wykonywalnego kodu, a także możliwość dzielenia danych z innymi przestrzeniami adresów. Programiści w asemblerze mają dostęp do przestrzeni danych dzięki rozwiniętym narzędziom przestrzeni adresowej.

By rozszerzyć możliwości adresowania wirtualnego bez konieczności modyfikacji formatu instrukcji, zastosowano zbiór 16 rejestrów Access Registers (AR). Każdy z tych rejestrów jest powiązany z odpowiadającym mu rejestrem General Purposes Register (GPR) w ten sposób, że gdy GPR jest używany jako rejestr bazowy dla lokalizacji operanda, odpowiadający mu AR jest używany do generowania przestrzeni danych, w której znajduje się operand. W ESA/370 Segment Table Origin zmieniono na Segment Table Destination (STD); źródło wybranej tablicy STD zależy od wartości 16 i 17 bitu słowa PSW. Może to być CR1 dla pierwotnej przestrzeni adresowej, CR7 dla drugiej oraz CR13 dla własnej przestrzeni adresowej lub zawartość rejestru dostępu translacji (ART), który generuje początek tablicy segmentu (Segment Table Origin) dla przestrzeni danych. ART jest układem niezbędnym do przeprowadzenia obliczeń przesunięcia adresowego przestrzeni danych (zob. rys. 6 i 7).

Wprowadzono specjalny rodzaj przestrzeni danych o wysokiej wydajności, zwanej hiperprzestrzenią, adresowanej jedynie blokami 4 KB, której strony mogą być przechowywane jedynie w pamięci rozszerzonej lub pomocniczej. By uzyskać dostęp do danych w hiperprzestrzeni, program musi przeczytać blok zawierający dane do jego przestrzeni adresowej. Jej działanie przypomina pamięć podręczną (cache) i zapewnia programowi bezpośredni dostęp do rozszerzonej pamięci o bardzo wysokiej wydajności. Dostęp do danych hiperprzestrzeni w pamięci rozszerzonej jest szybszy i prostszy niż dostęp do innego rodzaju urządzenia I/O; nie ma potrzeby otwierania zbioru danych, nie ma dodatkowych nakładów na metodę dostępu oraz nie ma potrzeby szeregowania losowych prób dostępu. Języki wysokiego poziomu takie jak FORTRAN lub COBOL mogą łatwo załadować dane z hiperprzestrzeni przy użyciu usług zwanych okienkowaniem. Podczas, gdy architektura XA usprawniła operacje I/O, celem powstania architektury ESA było pominięcie niektórych operacji I/O. Podsystem dynamicznego kanału odciąża zarządcę operacji I/O (IOS) od nakładu związanego z inicjacją operacji I/O (wybór ścieżki, inicjacja kolejkowania, itp.) lecz nie usprawnia samej operacji I/O. Czas operacji I/O przekracza czas odpowiedzi urządzenia. Średni nakład czasu procesora na inicjalizację instrukcji Start Subchannel (SSCH) wynosi około 10000 instrukcji oraz dodatkowo 2000 instrukcji niezbędnych do przełączania zadań. Podczas stronicowania rozkaz Resume Subchannel (RSCH) wymaga około 3000 instrukcji, a dodatkowy nakład na przełączanie zadań pozostaje ten sam. Dodatkowe opóźnienie musi być uwzględnione dla kolejkowania zarządcy IOS, kolejkowania ścieżki itp. Protokół początkowej selekcji zajmuje około 1 milisekundę przy kanałach równoległych oraz nieco dłużej przy kanałach ESCON.

Głównym powodem użycia hiperprzestrzeni było uniknięcie dodatkowych operacji oraz opóźnień podczas odzyskiwania danych; rezultatem był krótszy czas odpowiedzi dla pracy on-line i małych zadań wsadowych. Wielu twórców oprogramowania w IBM oraz inni dostawcy software'u docenili i wykorzystali zalety hiperprzestrzeni. Najbardziej popularne pakiety to CICS/ESA V3R1, CICS V2R1, TSO/E V2, DFSORT R11 oraz IMS/ESA, które wykorzystywały narzędzia przeszukiwania wirtualnego Virtual Lookaside Facility (VLF) do zarządzania przestrzenią danych oraz pamięcią rozszerzoną, procedury CLIST oraz REXX, ICF Catalog Entries oraz inne często ładowane moduły. VLF działa jako pamięć podręczna przechowująca operacje I/O, przetwarza dane jako obiekty i musi mieć ważną permanentną kopię na dysku (DASD).

W praktyce hiperprzestrzenie bywają rozwiązaniem niekorzystnym, gdyż rywalizacja o pamięć rozszerzoną między różnymi aplikacjami może powodować spadek wydajności systemu. Trzy parametry powinny być uważnie obserwowane:

- szybkość przenoszenia strony (z pamięci centralnej do rozszerzonej)
- wiek migracji oraz współczynnik migracji strony (który może być uzyskany z RMF)
- współczynnik przenoszenia dla pamięci rozszerzonej.

Współczynnik migracji strony

Współczynnik migracji strony określa raczej stopień użycia pamięci rozszerzonej niż jej rozmiar i wpływa to na czas przestoju jako wynik synchronicznego przesyłania danych. Na przykład:
łączny czas przestoju procesora = częstotliwość stronicowania na sekundę x czas przesyłania strony x liczba procesorów.

Zatem, czas przestoju dla procesora 3090-300J z częstotliwością stronicowania 600 stron na sekundę na procesor wynosi 600 x 70 x 104 x 3 = 0,126 sek. Wartość absolutna nie jest istotna o ile złożoność procesora nie przekracza jego ograniczeń pojemności. W tym przykładzie 300J straci około 8 MIPS (wg Gartner Group) z powodu operacji wymiany stron.

Z reguły jest tak, że dodatkowe nakłady poniesione na stronicowanie nie powinny wynosić więcej niż 5% wydajności systemu na procesor.

Wiek migracji

Wiek migracji określa średni czas, jaki strona przebywa w pamięci rozszerzonej bez żadnych odwołań do niej zanim zostanie ona przeniesiona do pamięci pomocniczej. Im wyższa wartość AUIC dla pamięci centralnej tym mniej konfliktów. Dwie trzecie tej wartości to czas w sekundach i wartość mniejsza niż 150 (100 sekund) wskazuje na poważny niedobór pamięci rozszerzonej. Kryteria wieku są wartościami ustawianymi w SYS1.PARMLIB dla różnych zastosowań pamięci rozszerzonej. Jeśli wartość wieku wymienianej strony jest wyższa niż kryteria wieku, system operacyjny zapisze stronę do pamięci rozszerzonej, w przeciwnym przypadku strona jest zapisywana bezpośrednio do pamięci pomocniczej, co może powodować pogorszenie czasu reakcji dla interaktywnych programów. Gdy poziom konfliktów o dostęp do pamięci rozszerzonej jest zbyt wysoki, hiperprzestrzenie będą rzadko z niej korzystały.

Współczynnik migracji strony

Współczynnik migracji strony wskazuje liczbę stron przeniesionych z pamięci rozszerzonej do pamięci pomocniczej. Algorytm LRU wybiera ramki do przeniesienia. Ponieważ ten krok nie może być wykonany bezpośrednio, strony są zapisywane z powrotem do pamięci centralnej a potem przy pomocy operacji I/O na dysk (DASD). Wysoki współczynnik migracji ma bardzo niekorzystny wpływ na wydajność systemu; procesory zostają zatrzymane podczas synchronicznego transferu danych operacji „page-in"; konflikty ramek w pamięci centralnej narastają i jeden z procesorów zostaje narażony na przeciążenie w rezultacie wielu operacji I/O oraz przełączania zadań.

Wartości domyślne parametrów nie powinny zezwolić optymalnym hiperprzestrzeniom oraz pamięci rozszerzonej na pełną eksploatację. Wiek migracji wynoszący 500 sekund powoduje zauważalny dodatkowy nakład przez nadmierną migrację do pamięci pomocniczej. MVS/ESA SP 4.2 rozszerza możliwości strojenia ustawiając priorytety dla wykorzystania pamięci rozszerzonej. Przy poprawnym strojeniu krytyczny wiek migracji może być obniżony do 400 sekund; preferencyjny sposób traktowania może być przyznany najważniejszym aplikacjom, lecz to zawsze odbywa się kosztem innych.

Rozwój architektury i sprzętu

Instrukcja Move Page pierwszy raz pojawiła się w modelach 3090 J, było także możliwe jej wykonanie na modelach S i kompatybilnych maszynach producentów PCM takich, jak Comparex 8/9X czy serii NAS/HDS EX. Instrukcja ta redukuje czas zużywany przez synchroniczne przesyły pomiędzy pamięciami centralną i rozszerzoną. Aczkolwiek IBM nie ogłosił oficjalnie nowych wartości czasowych, nowe procedury stronicowania zostały przypuszczalnie wpisane do mikroprogramu w celu zredukowania czasu pobierania instrukcji. Wszystko wskazuje na to, że wykonanie instrukcji Move Page trwało około 60 mikrosekund na modelu 3090 J oraz 9021-H0 (J hardware), tak więc poprawa nie była znacząca.

Pierwszy raz zastosowano instrukcję Move Page w HiperBatch, wykorzystując narzędzia wyszukiwania danych Data Lookaside Facility (DLF) do tworzenia i kontroli hiperprzestrzeni. Obiekt DLF jest tworzony tak jak jedna lub kilka hiperprzestrzeni w pamięci rozszerzonej jako rezultat pierwszego dostępu VSAM lub QSAM do wybranych zestawów danych rezydujących na dysku (DASD). Te zestawy danych mogą być dzielone przez działające współbieżnie albo następujące po sobie działania. Ładowane są one instrukcją OPEN oraz, o ile nie są zatrzymywane, usuwane podczas ostatniej instrukcji CLOSE. Produkt firmy IBM Pomocnik do Analizy HiperBatch'ów – HiperBatch Analysis Aid (HBAID) zaleca zarówno ubieganie się przez HiperBatch jak i rywalizowanie o dostęp do pamięci rozszerzonej. Przydział pamięci rozszerzonej wyłącznie dla HiperBatch może nasilić konflikty w pamięci.

Narzędzia asynchronicznego przepisania strony

Narzędzia te oferują poprawę wydajności dzięki możliwości przepisania strony z pamięci centralnej do rozszerzonej równolegle do pracy procesora. Operacja odwrotna, czyli przepisanie strony do pamięci centralnej nadal jest wykonywana synchronicznie. Udogodnienie to jest dostępne na komputerze 9021/H2 (opartym na 520 procesorach) oraz 9021/H5 (711 procesorów) i wkrótce będzie również dostępne na Comparex 9/9XX, 99/XXX oraz HDS seria GX.

Możliwe przyszłe drogi rozwoju

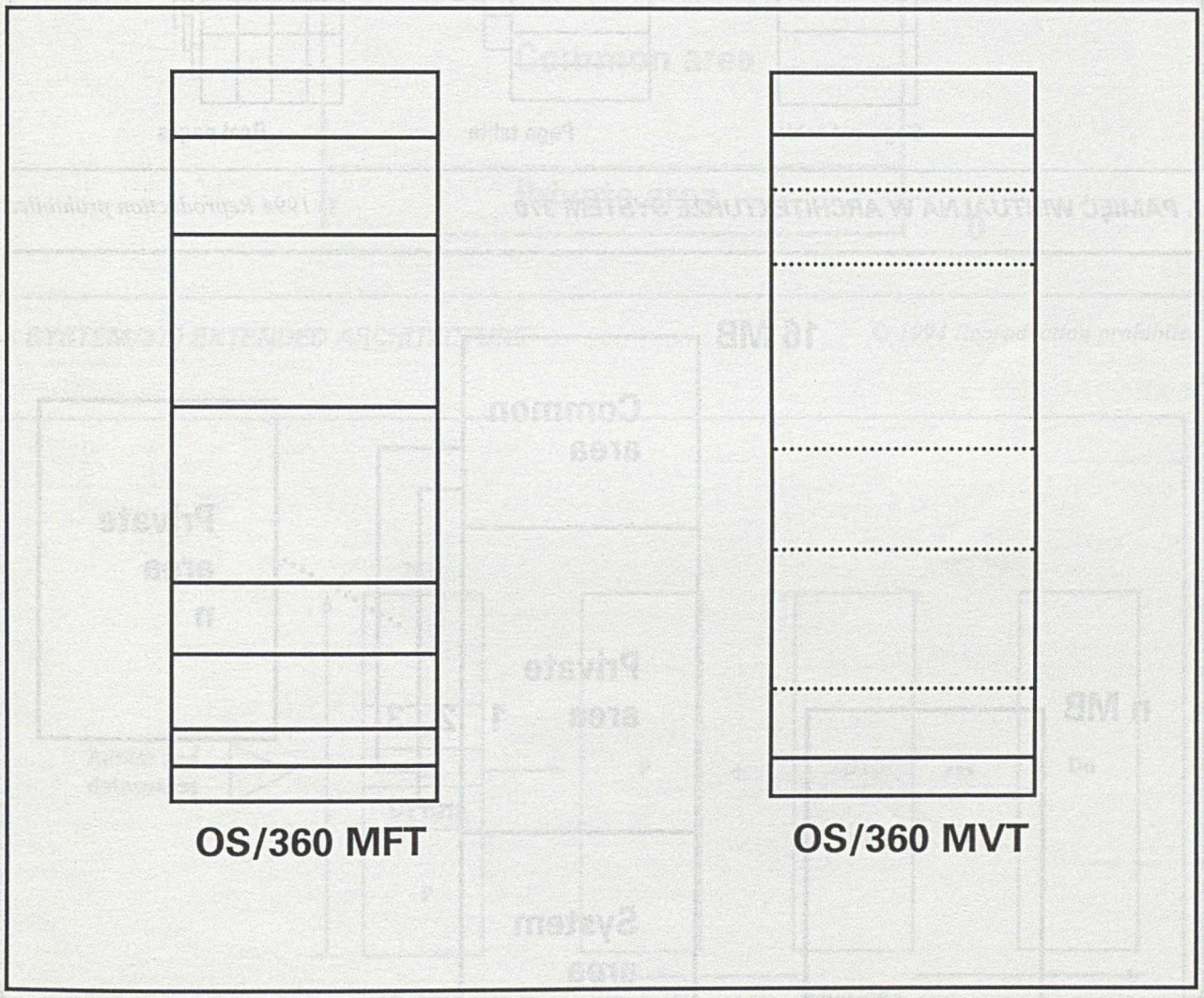
Największą wadą pamięci rozszerzonej jest brak bezpośredniego dostępu do urządzeń I/O, jej niestabilność, oraz fakt, że pomimo logicznego podziału nie może być współdzielona z innymi systemami. Struktura obecnych dużych systemów komputerowych (MAINFRAME) sugeruje, że zmiany sprzętowe pozwolą na bezpośrednie korzystanie z pamięci rozszerzonej, lecz będzie to wymagało zmian w architekturze polegających np. na dodatkowym bicie w Operation Request Block (ORB) lub w Channel Command Word (CCW) lub w obu naraz. Te zmiany w architekturze spowodują zróżnicowanie przyszłych modeli od obecnie istniejących, a czas ich wprowadzenia będzie zależny jedynie od decyzji marketingowych firmy IBM. W czasach recesji użytkownicy są skłonni przedkładać rozwiązania o lepszym współczynniku cena/wydajność nad najnowsze osiągnięcia techniki. Niemniej jednak zauważalny jest już koniec recesji i instalacje oparte o modele 9021 zbliżają się do rozmiarów uzasadniających poważne zmiany w architekturze.

Systemy ESA/390, ESA 92 oraz ESA 93 stanowią zestaw funkcji nie mających żadnego rzeczywistego wpływu na zarządzanie pamięcią. Inne ograniczenia będą mogły być rozwiązane dzięki dawno oczekiwanym samodzielnym układom półprzewodnikowym. Nie jest to łatwe zadanie, ponieważ wymagania czasowe dla bezpośrednio przyłączonych układów pamięci są znacznie bardziej krytyczne niż dla emulacji dysków SSD. Inne rozważania projektowe dotyczą priorytetów dostępu oraz szeregowania.

Wnioski

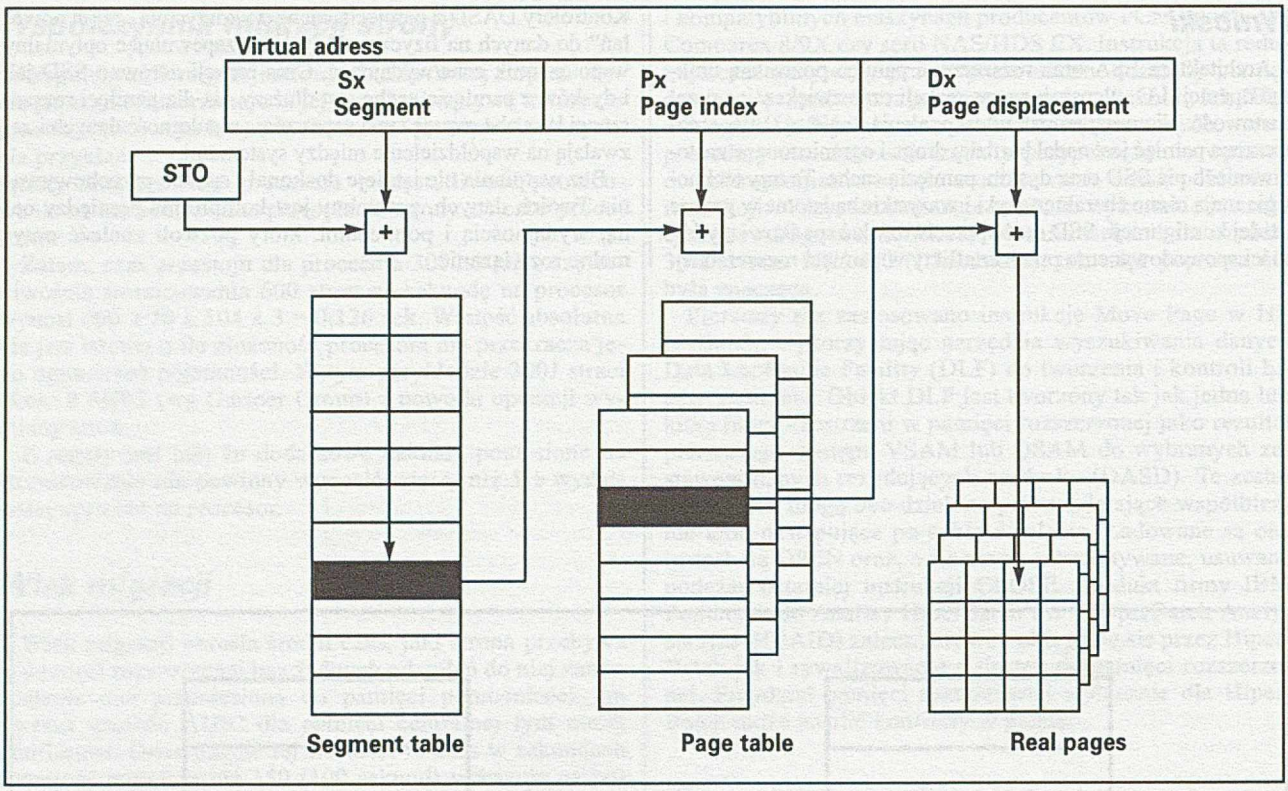
Architektura ESA oraz rozszerzona pamięć pozwalają uniknąć operacji I/O, ulepszając czas reakcji oraz zwiększając przepustowość. Niemniej jednak udoskonalenia te nie są tanie – rozszerzona pamięć jest nadal bardziej droga i ograniczona w zastosowaniach niż SSD oraz dyski z pamięcią cache. Te trzy technologie mają różne charakterystyki i wszystkie są istotne w prawie każdej konfiguracji. SSD może przeciwdziałać spadkowi wydajności spowodowanemu przez konflikty w pamięci rozszerzonej. Kontrolery DASD z pamięcią cache wykorzystują „rejon odwołań" do danych na fizycznych dyskach zapewniając optymalny współczynnik cena/wydajność. Czas reakcji zarówno SSD jak i dysków z pamięcią cache jest dłuższy niż dla pamięci rozszerzonej lecz oba rozwiązania zapewniają nieulotność danych i zezwalają na współdzielenie między systemami.

Bez wątpienia nie istnieje doskonały sposób przechowywania Twoich danych, potrzebny jest kompromis pomiędzy ceną, wydajnością i potrzebami, który pozwoli znaleźć optymalne rozwiązanie.



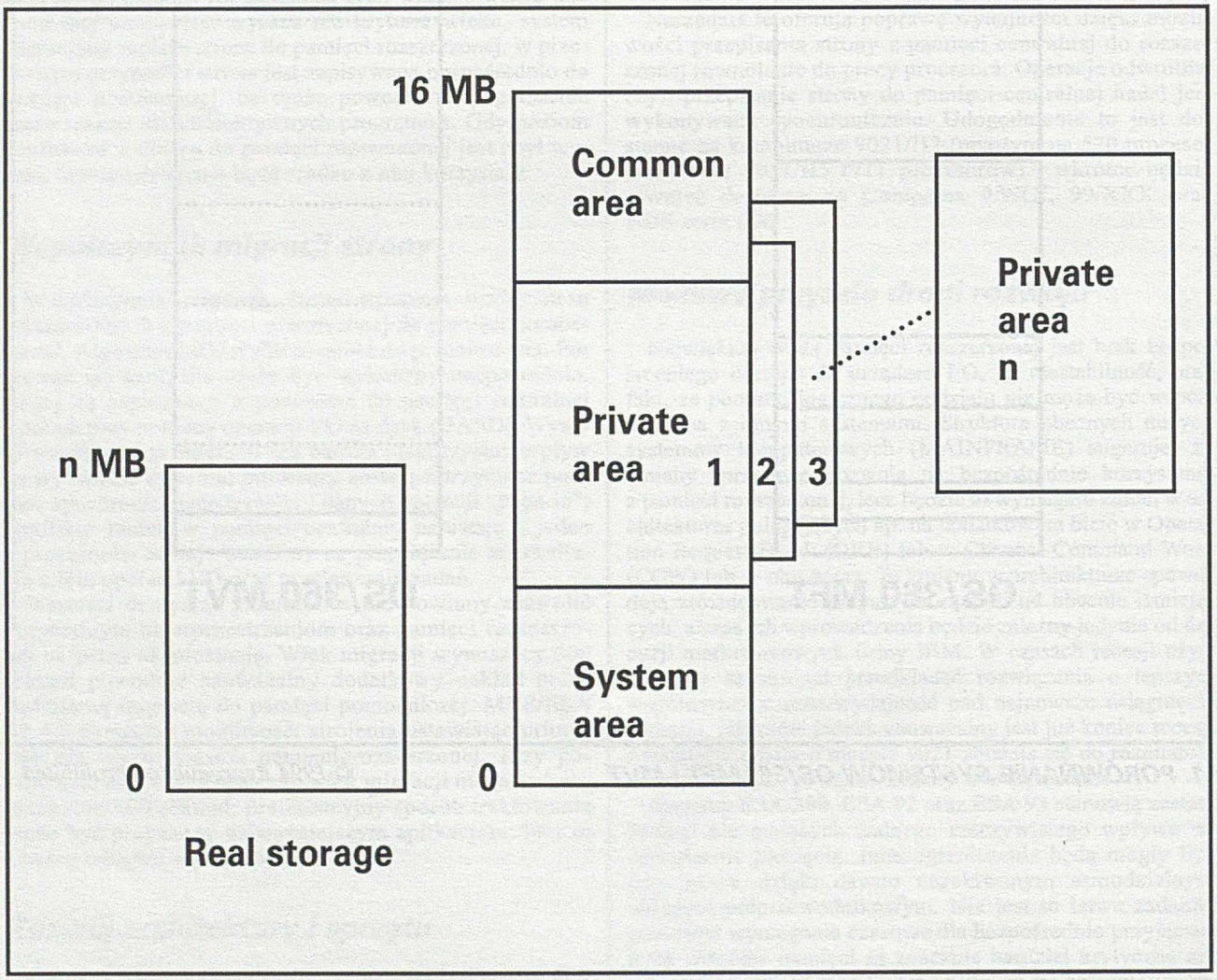
1. PORÓWNANIE SYSTEMÓW OS/360 MFT i MVT

© 1994 Reproduction prohibited



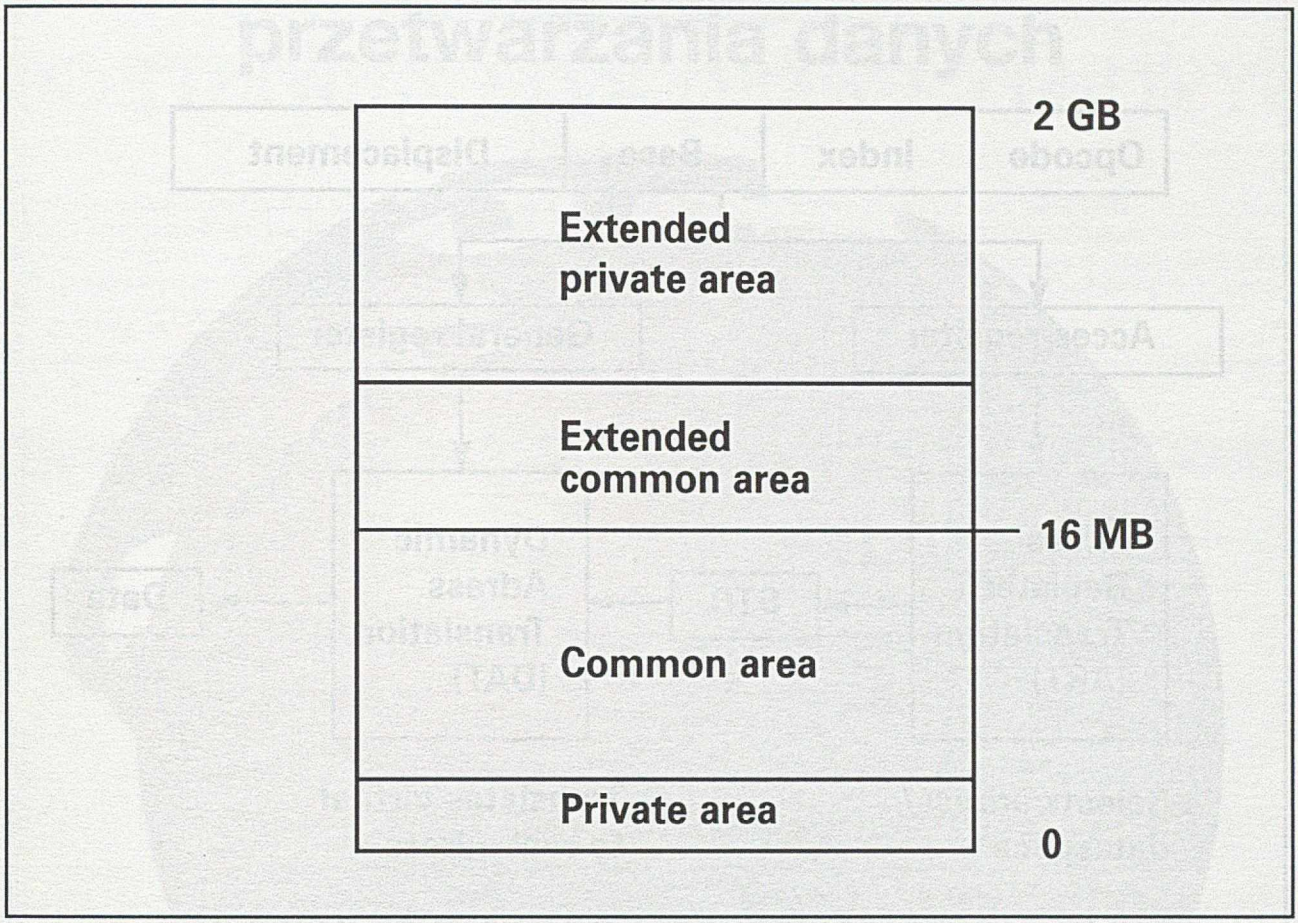
2. PAMIĘĆ WIRTUALNA W ARCHITEKTURZE SYSTEM 370

© 1994 Reproduction prohibited



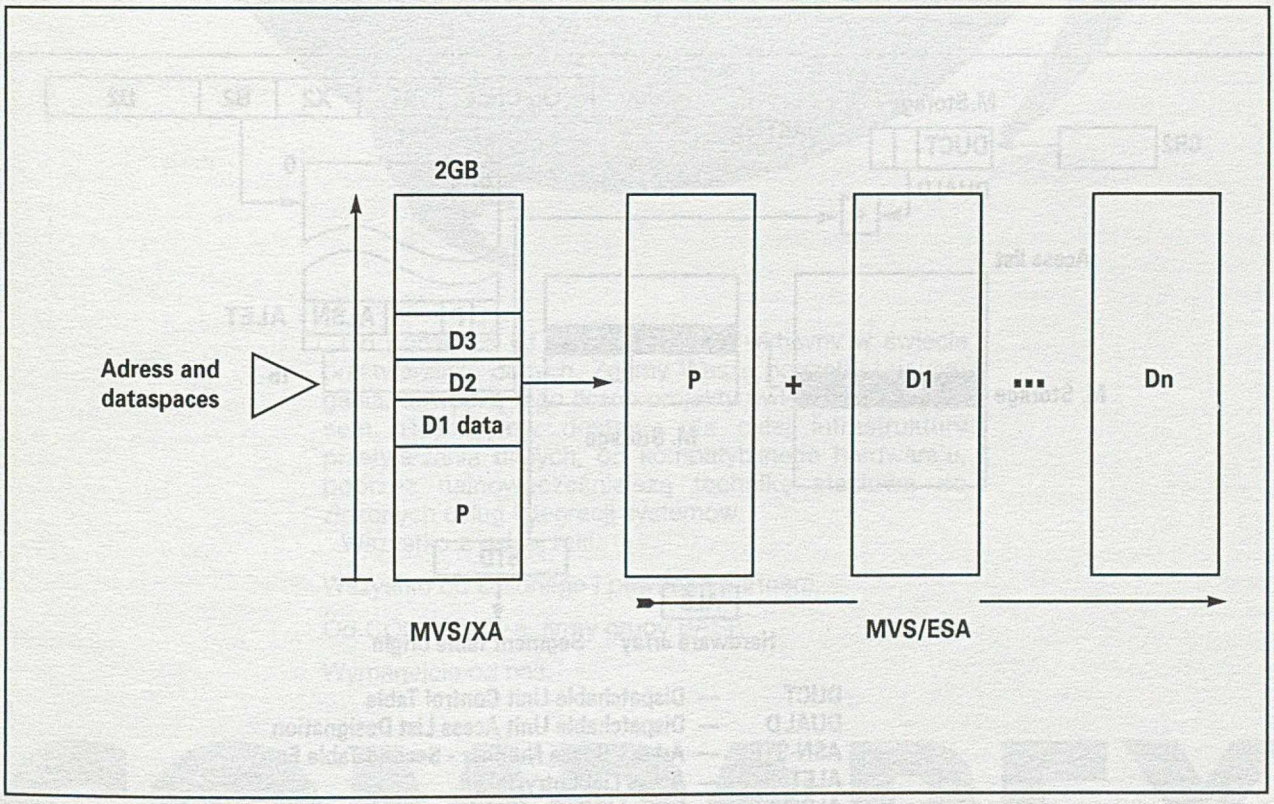
3. MVS i PAMIĘĆ WIRTUALNA

© 1994 Reproduction prohibited



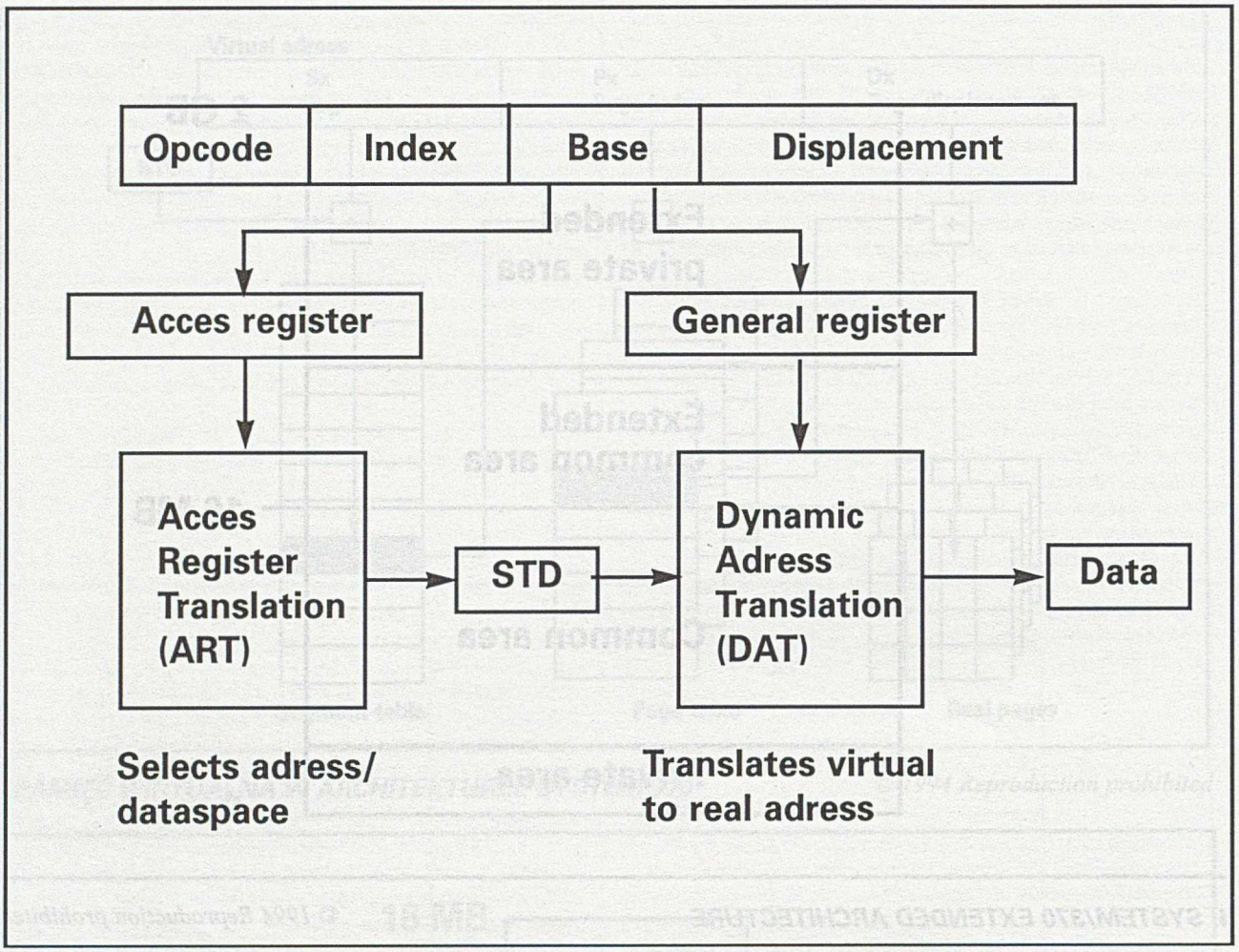
4. SYSTEM/370 EXTENDED ARCHITECTURE

© 1994 Reproduction prohibited



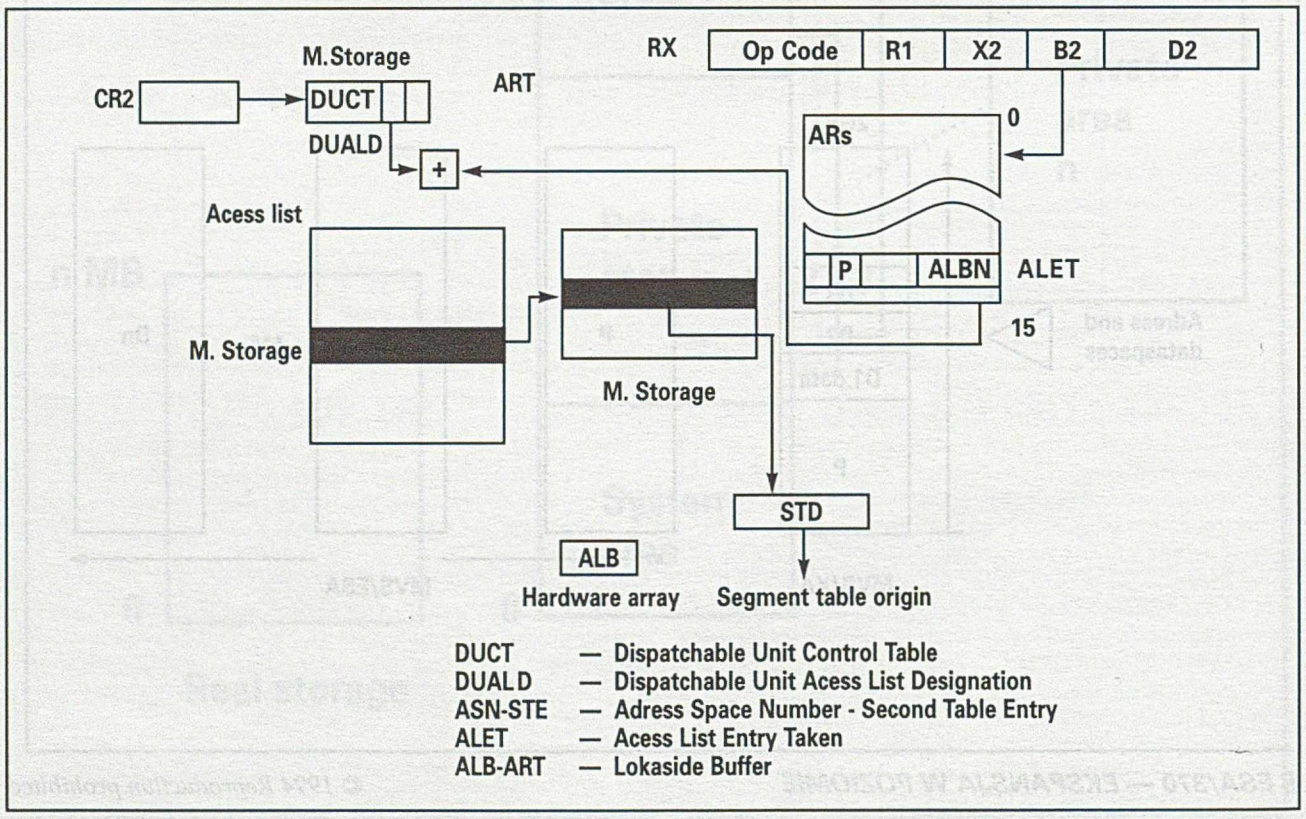
5 ESA/370 — EKSPANSJA W POZIOMIE

© 1994 Reproduction prohibited



6. ESA/370 — UŻYCIE REJESTRÓW DOSTĘPU

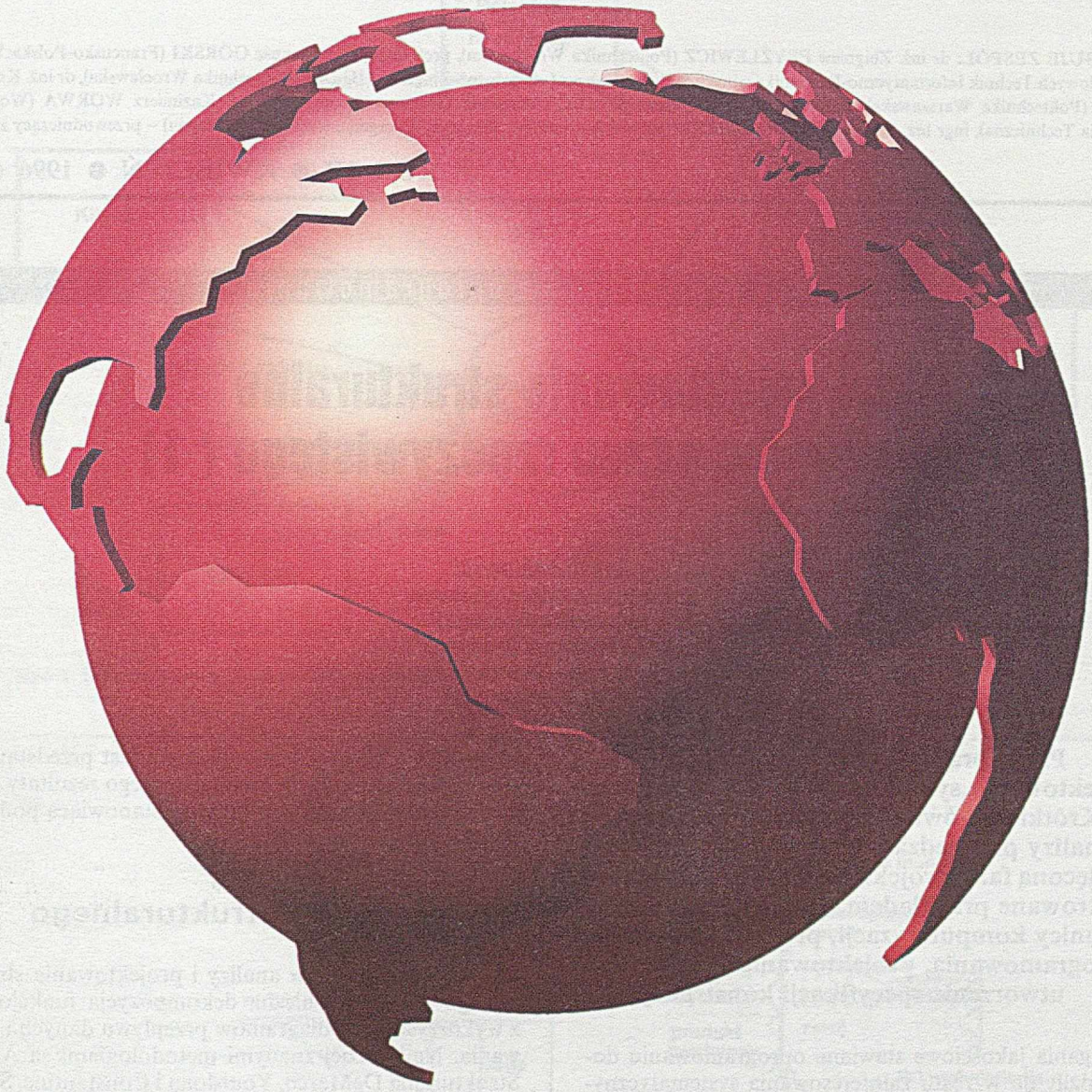
© 1994 Reproduction prohibited



7. ESA/370 — DZIAŁANIE ART

© 1994 Reproduction prohibited

Rozwiązania dla kompleksowego przetwarzania danych



Od ponad 25 lat COMPAREX jest aktywny w świecie przetwarzania danych. Znamy Wasze potrzeby i wymagania, dowodzą tego liczne projekty uwieńczone sukcesem. Realizujemy dostawy dla całej infrastruktury przetwarzania danych, od kompatybilnego hardware'u, poprzez najnowocześniejszą technikę sieciową do złożonych usług integracji systemów.
Wszystko z jednej ręki.

Wszystko od solidnego i pewnego partnera.

Od COMPAREX-a, firmy grupy BASF.

Wymagajcie od nas.

COMPAREX

COMPAREX Informationssysteme GmbH • Gottlieb-Daimler-Straße 10 • 68165 Mannheim
Telefon 0621/4009-201 • Fax 0621/4009-409 • e-Mail: 101346,3707 (Compuserve)

01/96

publikacje

SYSTEMY CZASU RZECZYWISTEGO

REDAGUJE ZESPÓŁ: dr inż. Zbigniew FRYŹLEWICZ (Politechnika Wrocławska), prof. dr hab. inż. Janusz GÓRSKI (Francusko-Polska Wyższa Szkoła Nowych Technik Informatyczno-Komunikacyjnych w Poznaniu), prof. dr hab. inż. Zbigniew HUZAR (Politechnika Wrocławska), dr inż. Krzysztof SACHA (Politechnika Warszawska), prof. dr hab. inż. Tomasz SZMUC (Akademia Górniczo-Hutnicza), dr inż. Kazimierz WORWA (Wojskowa Akademia Techniczna), mgr inż. Zdzisław ŻURAKOWSKI (Instytut Automatyki Systemów Energetycznych we Wrocławiu) – przewodniczący zespołu.

ROK III ● KWIECIEŃ ● 1996 ● 17

Metody projektowania oprogramowania

Projektowanie strukturalne systemów czasu rzeczywistego (1)

Jan Kwiatkowski
Wydziałowy Zakład Informatyki
Politechnika Wrocławska
Stanisław Szejko
Zakład Zastosowań Informatyki
Politechnika Gdańska

Praca przedstawia strukturalną metodę projektowania systemów czasu rzeczywistego. Krótkie omówienie przebiegu i rezultatów analizy poprzedza właściwą część artykułu, poświęconą fazie projektowania. Kolejne kroki, ilustrowane przykładem, obejmują określenie granicy komputeryzacji, projekt środowiska i oprogramowania, projektowanie plików oraz utworzenie specyfikacji konstrukcyjnej.

Wymagania jakościowe stawiane oprogramowaniu doprowadziły do wzrostu zainteresowania systematycznymi metodami jego wytwarzania. Ma to szczególne znaczenie w przypadku systemów pracujących w czasie rzeczywistym i wymagających zapewnienia bezpieczeństwa. W konsekwencji pojawiło się wiele nowych metodologii systematycznej konstrukcji oprogramowania, takich jak: prototypowanie [5, 25], podejście zorientowane na struktury danych [1, 18], podejście strukturalne [7, 11, 18, 21, 27, 29], podejście obiektowe [2, 20], metodologie formalne [3, 22, 25, 28].

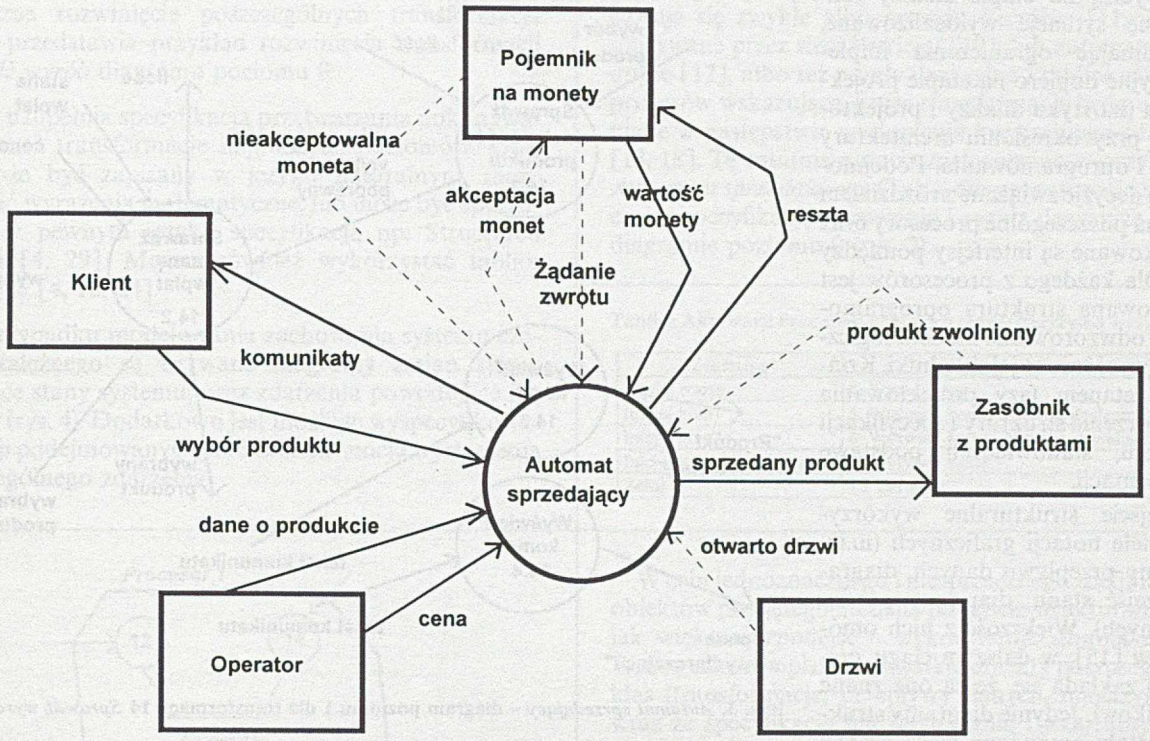
Wspólną cechę wymienionych podejść stanowi wykorzystanie graficznych notacji do modelowania wymagań, struktury i zachowania projektowanego systemu oraz wspomaganie procesu analizy, projektowania i konstrukcji narzędziami typu CASE [17].

Szczególną pozycję wśród podejść systematycznych, z uwagi na powszechność ich stosowania, zajmują metodologie strukturalne. Idea podejścia strukturalnego, ze szczególnym uwzględnieniem fazy analizy, została szczegółowo przedstawiona w artykułach zamieszczonych uprzednio w *Informatyce* [15, 21]. Stąd niniejsza praca nie zawiera omówienia konkretnych metodologii strukturalnych ani stosowanych przez nie notacji, nie omawia też szczegółowo fazy analizy – jej celem jest przedstawienie etapu projektowania, przekształcającego rezultaty analizy – w specyfikację konstrukcyjną, stanowiącą podstawę implementacji.

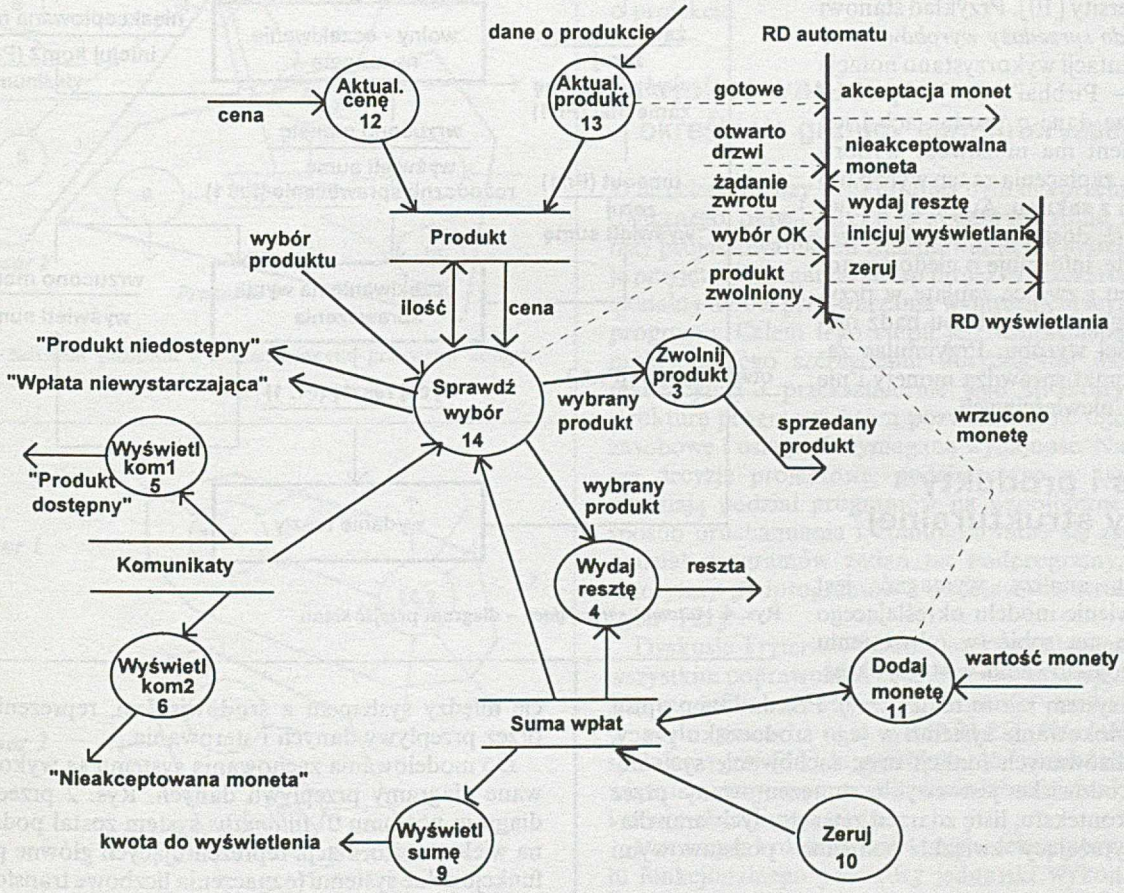
Idea podejścia strukturalnego

Ideą leżącą u podstaw analizy i projektowania strukturalnego jest hierarchiczna dekompozycja funkcjonalna z wykorzystaniem diagramów przepływu danych i sterowania. Najbardziej znanymi metodologiami są Analiza Strukturalna DeMarco, Yourdona i Constantine, SA/SD Yourdona, SSADM [7,23,29], SA/SD/RT Warda i Mellora oraz metodologia Hatley'a-Pirbhai'a [13,18,27].

Podstawowym założeniem tworzenia systemu według podejścia strukturalnego jest utworzenie logicznego modelu wymagań, na podstawie którego jest tworzony następnie system. W pierwszym kroku konieczne jest zdefiniowanie celu stawianego przed projektowanym systemem. Jego określenie umożliwia przejście do fazy analizy, rozpoczynającej się od zdefiniowania *modelu środowiska.* Model ten określa zachowanie systemu z punktu widzenia jego zewnętrznego otoczenia. Obejmuje on wyraźne zakreślenie granic systemu, opis zdarzeń zewnętrznych, na które system powinien reagować, oraz powiązania z współpracującymi z nim obiektami. Kolejnym krokiem jest stworzenie *modelu zachowania,* czyli opisu zachowania systemu w odpowiedzi na poszczególne zdarzenia. Na tym etapie dokonuje się opisu transformacji, wykonywanych w odpowiedzi na wystąpienia zdarzeń, oraz specyfikuje się wymagane informacje (przepływy, struktury danych). Model uzupełnia opis przepływu sterowania, określający dynamikę systemu. Stworzenie tych dwóch modeli praktycznie kończy fazę analizy problemu.



Rys. 1. Schemat kontekstu *Automatu sprzedającego*

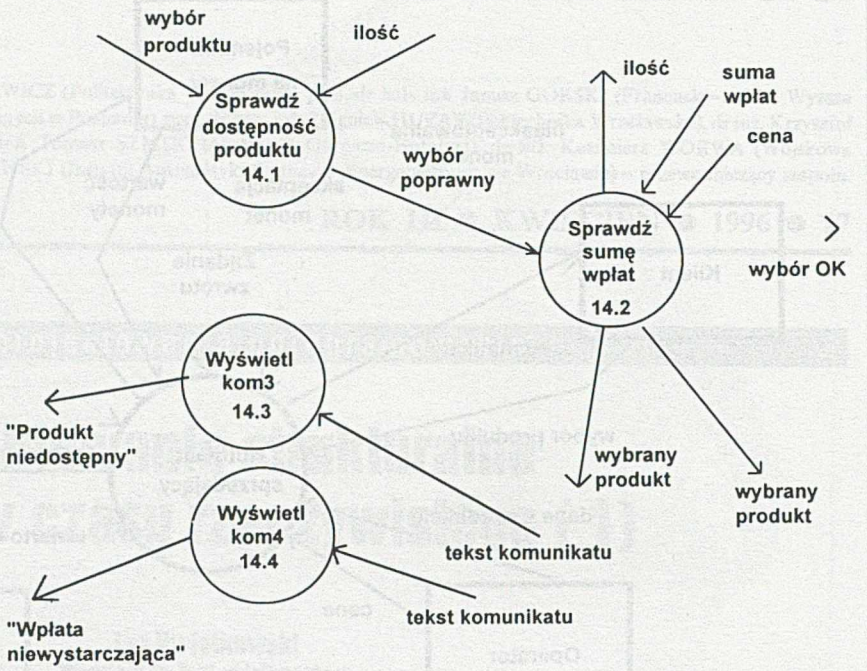


Rys. 2. *Automat sprzedający* – diagram poziomu 0 z uwidocznioną specyfikacją sterowania

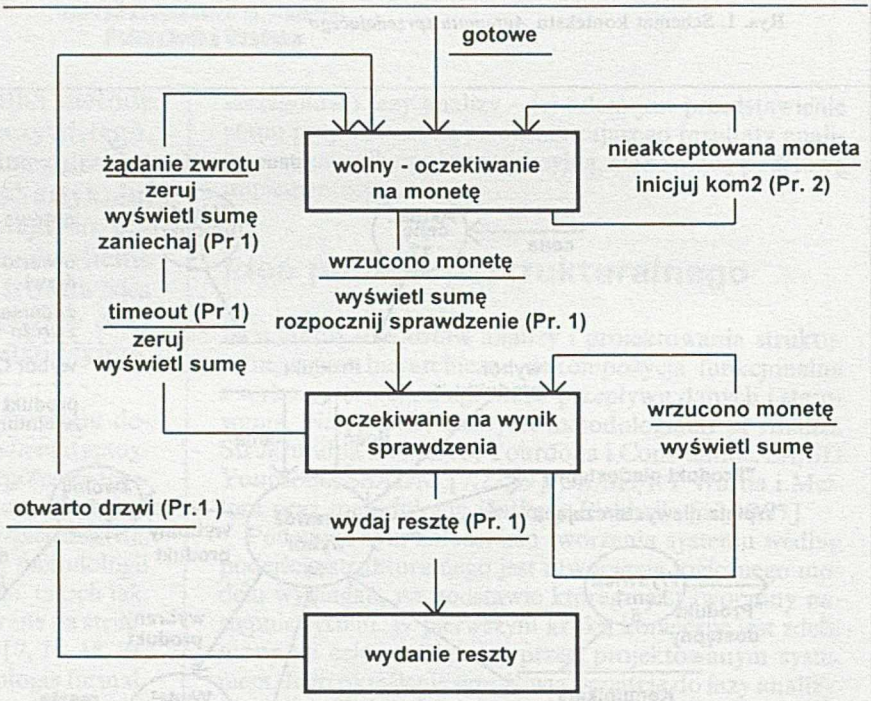
Zazwyczaj na etapie analizy rozważa się sytuacje wyidealizowane, uwzględniając ograniczenia implementacyjne dopiero na etapie projektowania (na styku analizy i projektowania), przy określaniu architektury sprzętu i oprogramowania. Podejmowane są decyzje związane z rozdziałem zadań na poszczególne procesory oraz specyfikowane są interfejsy pomiędzy nimi. Dla każdego z procesorów jest projektowana struktura oprogramowania i odwzorowanie modelu logicznego w wyróżnione jej składniki. Końcowym etapem fazy projektowania jest stworzenie struktury i specyfikacji programu, stanowiących podstawę implementacji.

Podejście strukturalne wykorzystuje wiele notacji graficznych (m.in. diagramy przepływu danych, diagramy przejść stanu, diagramy powiązań danych). Większość z nich omówiono w [15]; w dalszym ciągu prezentacji zakłada się, że są one znane czytelnikowi. Jedynie diagramy struktury zostały omówione nieco szerzej w p. 3.5.

Fazę projektowania ilustruje przykład, będący kontynuacją analizy wykonanej przez R. Farmera z De Montfort University [10]. Przykład stanowi *Automat do sprzedaży wyrobów*, a do jego prezentacji wykorzystano notację Hatley'a – Pirbhai'a [13]. Operator wprowadza dane o produktach i ich cenie. Klient ma możliwość wyboru produktu, zapłacenia za ten wyrób lub rezygnacji z zakupu. Automat sprawdza zapłatę, dostarcza produkt zwracając resztę, informuje o niedostępności wyrobu i zwraca zapłatę w przypadku rezygnacji z zakupu bądź niedostępności wyrobu. Przyjmując zapłatę automat sprawdza monety i nie akceptuje niewłaściwych.



Rys. 3. *Automat sprzedający* – diagram poziomu 1 dla transformacji 14 *Sprawdź wyrób*



Rys. 4. *Automat sprzedający* – diagram przejść stanu

Zakres i produkty analizy strukturalnej

Zadaniem analizy wymagań jest przedstawienie modelu określającego *co* system ma robić (w odróżnieniu od fazy projektowania, w której określa się *jak* system ma to realizować). Przedmiotem opisu są więc: ulokowanie systemu w jego środowisku pracy, zbiór realizowanych funkcji oraz zachowanie systemu. Model środowiska jest zwykle reprezentowany przez schemat kontekstu, listę zdarzeń zewnętrznych oraz diagram, wyrażający związki pomiędzy podstawowymi obiektami.

Schemat kontekstu *Automatu sprzedającego* przedstawia rys. 1. Zostało na nim uwidocznione środowisko systemu, reprezentowane przez terminatory, oraz interakcje między systemem a środowiskiem, reprezentowane przez przepływy danych i sterowania.

Do modelowania zachowania systemu są wykorzystywane diagramy przepływu danych. Rys. 2 przedstawia diagram poziomu 0 *Automatu*: system został podzielony na wiele transformacji, reprezentujących główne procesy funkcjonalne systemu (oznaczenia liczbowe transformacji za [10]). Wyspecyfikowane są przepływy danych, wiążące transformacje, oraz magazyny (zbiory), reprezentujące logiczną strukturę przechowywanych przez system danych.

Dalsze uszczegółowienie funkcji systemu stanowi hierarchiczne rozwinięcie poszczególnych transformacji. Rys. 3 przedstawia przykład rozwinięcia transformacji *Sprawdź wyrób* diagramu poziomu 0.

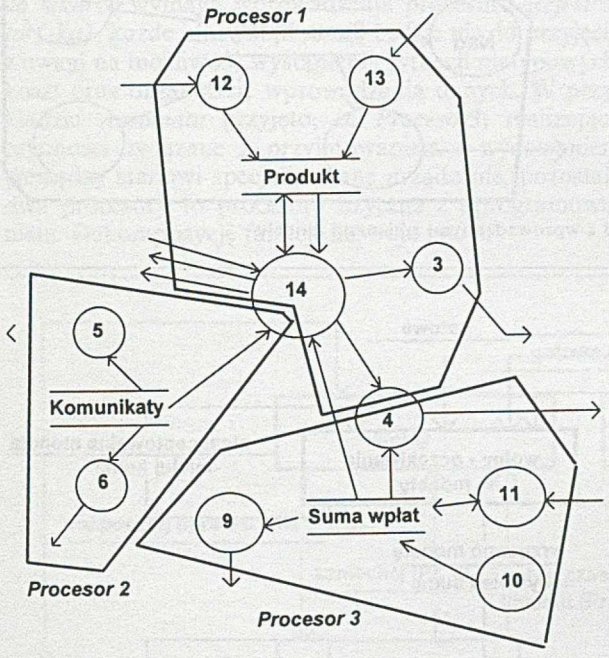
Opis uzupełnia specyfikacja przetwarzania dokonywanego przez transformacje najniższych poziomów. Opis ten może być zapisany w języku naturalnym, może zawierać wyrażenia matematyczne, lub może być sporządzony w pewnym języku specyfikacji, np. Structured English [4, 29]. Można również wykorzystać tablice decyzyjne [4, 16, 24].

W przypadku modelowania zachowania systemu czasowo zależnego są używane diagramy zmian stanu, opisujące stany systemu, oraz zdarzenia powodujące ich zmiany (rys. 4). Dodatkowo jest możliwe wyspecyfikowanie akcji podejmowanych jako konsekwencja wystąpienia poszczególnego zdarzenia.

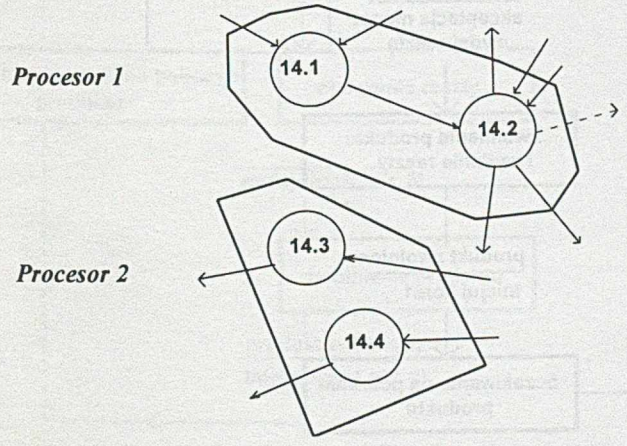
Do opisu przepływu sterowania i reakcji na zdarzenia stosuje się zwykle albo procesy sterujące [11, 27, 29], opisywane przez sieci Petriego [19], lub automaty skończone [12], albo też reguły decyzyjne i tablice aktywacji procesów wskazujące, które transformacje będą uaktywniane w następstwie wystąpienia konkretnego zdarzenia [13, 18]. Tę ostatnią notację zastosowano w przykładzie *Automatu sprzedającego* (Tab.), dla uproszczenia umieszczając specyfikację sterowania i reguły decyzyjne RD na diagramie poziomu 0 (rys. 2).

Tablica Aktywacji Procesów opisująca regułę decyzyjną wyświetlania

Zdarzenie	Transformacja
Wydaj resztę	4. Wydaj resztę
Inicjuj kom1	5. Wyświetl kom1 „Produkt dostępny"
Inicjuj kom2	6. Wyświetl kom2 „Nieakceptowalna moneta"
Inicjuj wyświetlenie sumy	9. Wyświetl sumę
Zeruj	10. Zeruj



Rys. 5a. Schemat poziomu 0 z zaznaczonymi granicami komputeryzacji



Rys. 5b. Poziom 1 procesu 14 – przydzielenie do procesorów

W celu jednoznacznego i niesprzecznego zdefiniowania obiektów przyjętego modelu, podejście strukturalne – tak jak większość podejść – wykorzystuje *słownik danych*. W słowniku są opisane wszystkie obiekty poszczególnych klas (transformacje, przepływy danych, magazyny itd.) wraz ze specyfikacją ich zachowania i wzajemnymi powiązaniami. Umożliwia to łatwą weryfikację kompletności i stopniowe rozwijanie modelu. Przy korzystaniu z narzędzi CASE wspierających proces analizy i projektowania systemów, słownik danych przyjmuje postać repozytorium projektowego, przechowującego całą wiedzę o projekcie.

Projektowanie – określenie granicy komputeryzacji

O ile celem analizy strukturalnej było utworzenie modelu logicznego, określającego *co* system ma robić, zadaniem fazy projektowania jest zdefiniowanie, *jak* system realizuje przyjęte wymagania. „Etap projektu przekształca funkcjonalny model przetwarzania w implementacyjny model programu. Celem tego etapu jest uzupełnienie modelu funkcjonalnego szczegółami dotyczącymi metodologii realizacyjnej i przekształcenie jego struktury w taką strukturę programu, która pozwoli spełnić ograniczenia zasobowe i osiągnąć wymaganą wydajność. Najważniejsze decyzje projektowe, podejmowane w tym etapie, określają podział programów na współbieżne zadania, sposób uruchamiania i komunikowania się zadań oraz podział programów zadań na podprogramy, a także algorytmy podprogramów i podstawowe struktury danych" [21].

Dyskusję kryteriów jakości oprogramowania, przede wszystkim poprawności, bezpieczeństwa, weryfikowalności, modyfikowalności i przenośności można znaleźć w wielu pracach, m.in. [8,18,21,25]. Warunkiem ich spełnienia jest [18] hierarchiczność i modularność rozwiązania, przy zapewnieniu niezależności funkcjonalnej i minimalizacji powiązań.

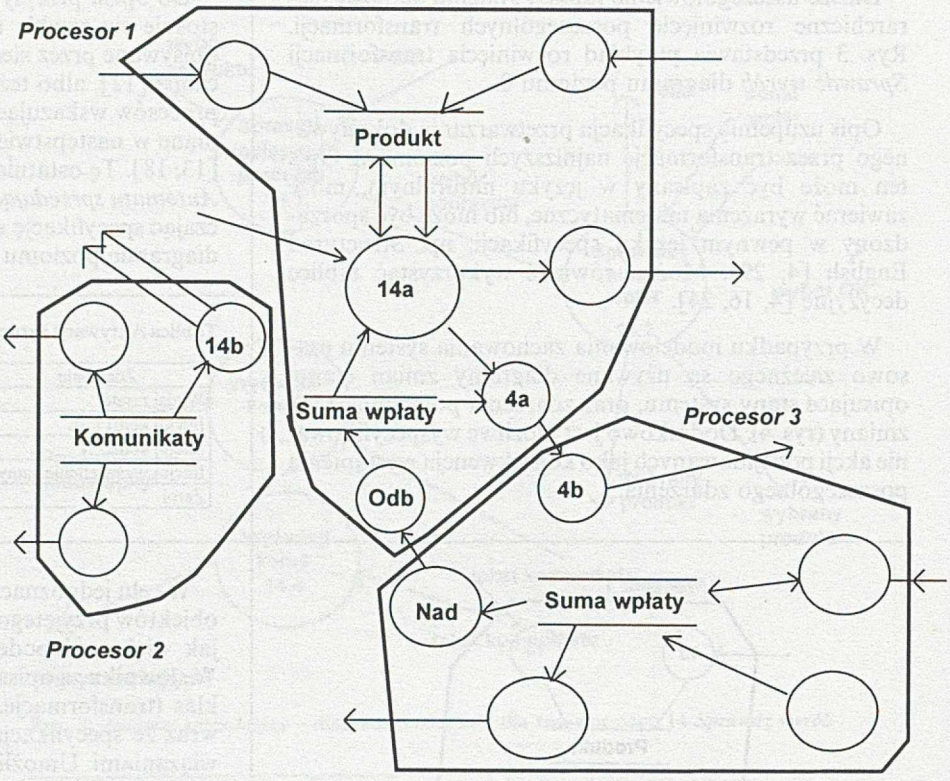
Pierwszym krokiem projektu jest dekompozycja modelu funkcjonalnego pomiędzy jednostki wykonujące zaplanowane przetwarzania, nazywane *procesorami*. Produkty tej fazy nazywane są też *modelem procesorowym* [11, 27, 29]. Obok programowanych procesorów rolę tę

mogą pełnić ludzie, jak np. operator systemu, oraz specjalizowane układy sprzętowe, dedykowane do realizacji wybranych transformacji. W trakcie określania *granicy komputeryzacji* wskazany zostaje podzbiór funkcji wykonywanych przez oprogramowanie poszczególnych komputerów systemu i oddzielony od zakresu czynności przewidywanych dla ludzi i funkcji spełnianych przez dedykowane urządzenia. Ustalenie granicy komputeryzacji przeplata się z koniecznością właściwego doboru procesorów, wymaga też uwzględnienia współbieżności w wykonywaniu zadań oraz istniejących ograniczeń implementacyjnych. Jednocześnie, określenie granicy pozwala na jednoznaczne wskazanie zbioru nieodzownych interfejsów systemu.

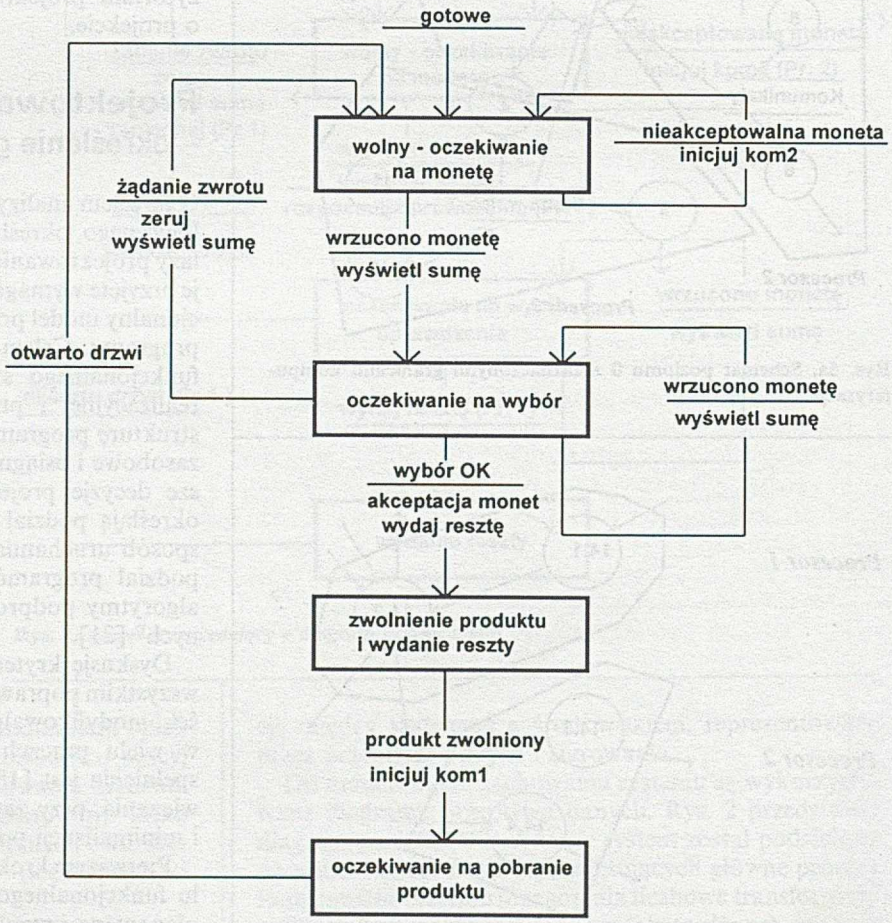
Wygodnym sposobem wyznaczenia granicy komputeryzacji jest narysowanie jej na diagramie przepływu danych możliwie wysokiego poziomu, tak jak to ilustruje rys. 5a. Zakreślona granica lokuje transformacje i magazyny poziomu 0 do trzech grup: *Procesora1*, nadzorującego pracę automatu, *Procesora2*, zapewniającego wyświetlanie informacji na panelu oraz *Procesora3*, obsługującego czynności związane z przyjmowaniem i wydawaniem pieniędzy. Granica przecina transformację 14 *Sprawdź wybór*, co oznacza, że część jej procesów składowych przydzielona została do *Procesora1*, inne (związane z wyświetlaniem komunikatów) – do *Procesora2*. Znajduje to swoje odbicie na rys. 5b.

Tylko pozornie zadanie wyznaczenia granicy komputeryzacji jest czynnością łatwą. Błędne zdefiniowanie granicy może prowadzić do niespełnienia wymagań, bądź znacznego skomplikowania konstrukcji. Poniżej omówione zostały pokrótce najważniejsze aspekty podejmowanych decyzji, szczególnie istotnych w przypadku systemów pracujących w czasie rzeczywistym.

Dobór procesorów. Decyzja o wyborze procesorów jest wypadkową wielu czynników: *potrzeb przetwarzania*, wynikających ze zdefiniowanego modelu zachowania – pod uwagę bierze się przewidywaną wielkość kodu, klasy wykonywanych zadań, złożoność obliczeń, zakładane wielkości danych, ich przepływy i preferowane nośniki; *potrzeb komunikacji*, wynikających z zadań systemu, z oszacowania wielkości i częstotliwości przepływów danych, narzucanych także przez prowadzoną przez użytkownika politykę IS/IT (ang. *Information Strategy/ /Information Technology*); *wymaganej*



Rys. 6. Granice komputeryzacji z wprowadzonymi zmianami modelu



Rys. 7a. Diagram przejść stanu dla procesów *Procesora 3* (przyjmującego monety)

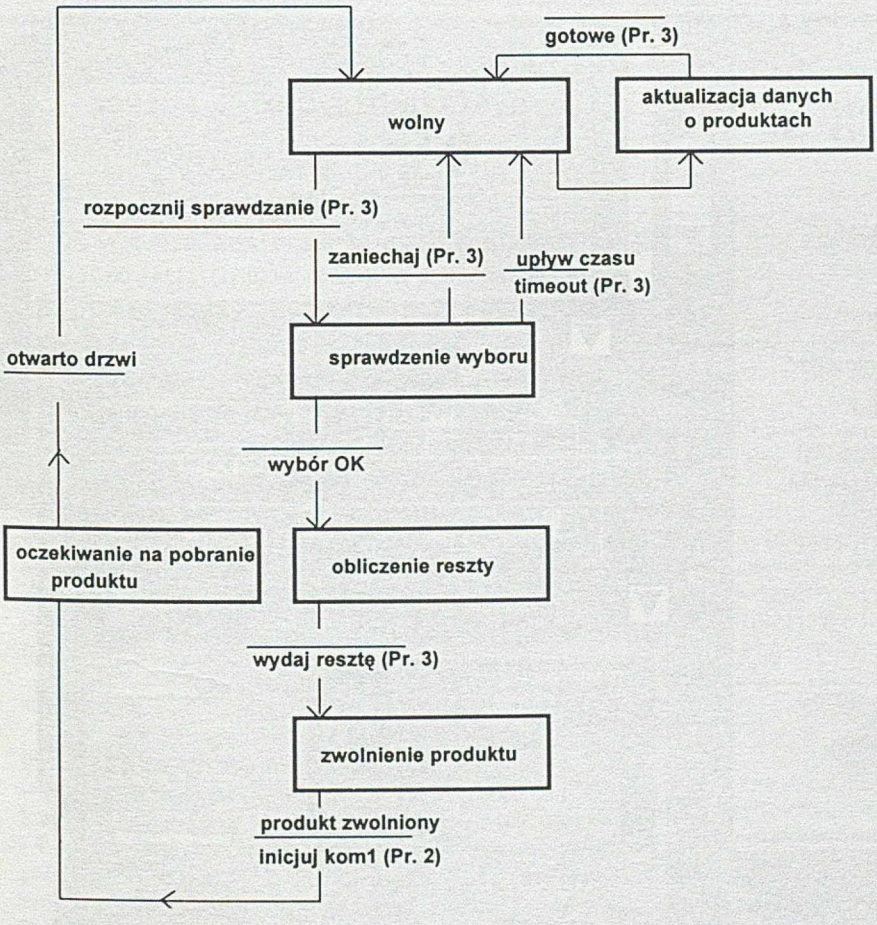
współbieżności i efektywności przetwarzania zadań przez procesory; *ograniczeń implementacyjnych i zdefiniowanych wymagań*, jak np. żądania, by w systemie wykorzystać już istniejący sprzęt; *właściwości procesorów*, dostępnych do wykorzystania w systemie, m.in. ich efektywności, niezawodności, kosztu oraz przystosowania do współpracy z innymi jednostkami. Proponowany w przykładzie dobór procesorów ma charakter demonstracyjny, zbiór zadań *Automatu sprzedającego* nie wymaga wykorzystania aż trzech procesorów.

Dekompozycja zadań między oprogramowanie, ludzi i dedykowany sprzęt. Przy analizie, które z procesów objąć komputeryzacją, należy wziąć pod uwagę korzyści i nakłady, jakie pociąga za sobą każdorazowo podjęta decyzja. Często przyjmuje się, że transformacje heurystyczne, decyzyjne, są realizowane przez ludzi, a transformacje o dobrze zdefiniowanych algorytmach i podlegające częstym modyfikacjom – przez oprogramowanie. Na przykład, w typowym systemie wspomagania obsługi gabinetu lekarskiego, transformacja *Rejestruj pacjentów do lekarza* wymaga wprowadzenia procesora *Rejestrator(-ka)*, każde inne rozwiązanie będzie nie do przyjęcia z uwagi na możliwość wystąpienia sytuacji nietypowych, koszt oraz organizację wprowadzania danych. W przykładzie *Automatu* przyjęto, że *Procesor3*, realizujący czynności związane z przyjmowaniem i wydawaniem pieniędzy stanowi specjalizowane urządzenie, pozostałe dwa procesory to procesory fizyczne z oprogramowaniem. Dekompozycję funkcji ilustruje rys. 5.

Transformacje wykonywane przez ludzi dają podstawę do zdefiniowania reguł postępowania operatorów systemu, a te, które wyodrębniono do realizacji przez specjalizowane układy – do określenia wymagań na działanie sprzętu.

Dostępność danych, ich współużytkowanie lub wyłączność, to czynniki wpływające na dekompozycję zadań między procesory i podjęcie decyzji o rozmieszczeniu zbiorów danych. W sytuacji, gdy dostęp do zbioru jest nieodzowny dla procesów więcej niż jednego procesora, jest możliwych kilka strategii postępowania:

- Umieszczenie danych na nośniku zewnętrznym w stosunku do przyjętych procesorów. Będzie to wymagało organizacji zarządzania tak wydzielonym zasobem.
- Umieszczenie zbiorów w jednym z procesorów. Oznacza to konieczność zapewnienia dostępu procesom innych procesorów, a więc dodanie transformacji udostępniających i/lub przyjmujących dane.
- Umieszczenie części danych w jednym z procesorów, a części – w drugim. Zmniejsza to czas dostępu do grupy danych i częstotliwość odwołań do innego procesora, wymaga jednak organizacji dostępu, analogicznie jak poprzednio.
- Powielanie danych w procesorach z nich korzystających – rodzące obok problemów komunikacji i dostępu także problemy zapewnienia spójności danych. Tak postąpiono z danymi o wpłaconych monetach, zlokalizowanymi w *Procesorze3*, przechowywanymi także w *Procesorze1* (rys. 6).



Rys. 7b. Diagram przejść stanu dla procesów *Procesora 1* (sterującego wydawaniem produktu)

Dekompozycja sterowania. Rozdzielenie transformacji między procesory pociąga za sobą konieczność określenia sterowania w każdym z nich (pozostawienie sterowania w jednym z procesorów oznaczałoby ich patologiczne powiązanie [7], a jego rezultatem byłyby sygnały „włącz-wyłącz" pomiędzy procesorami).

Rysunek 7 przedstawia diagramy przejść stanów jakie przyjęto dla *Procesorów1* i *3* przy dekompozycji sterowania *Automatu* (diagram z rys. 4); nie zostały zamieszczone reguły decyzji przyjęte dla procesorów i odpowiadające im tablice aktywacji. Diagram ilustruje dwa związane z dekompozycją problemy:

- Konieczność synchronizacji zachowania procesorów w sytuacji ich współbieżnego działania. W szczególności zaowocowało to wprowadzeniem sygnału **zaniechaj,** wysyłanego przez *Procesor1* po identyfikacji sygnału **żądanie zwrotu** (pieniędzy), a powodującego w *Procesorze3* przerwanie sprawdzania i powrót do stanu początkowego.
- Współpracujące z procesorem otoczenie nabiera cech fizycznych a przyjętym w fazie analizy wyidealizowanym wyobrażeniom nadaje się cechy praktycznej implementacji. Stąd w przykładzie wprowadzono limit czasu oczekiwania na wy-

bór produktu przez klienta, a także przerwanie oczekiwania na wynik sprawdzenia w *Procesorze1* na skutek sygnału *timeout*. Opis sformalizowanej procedury podziału specyfikacji sterowania między procesory można znaleźć w [27].

Interfejsy. Podział modelu funkcjonalnego między odrębne procesory spowoduje wystąpienie konieczności zapewnienia komunikacji pomiędzy nimi, jak również takiego ich sprzęgnięcia, by pożądane dane i funkcje były dostępne we właściwym czasie i w pożądanej formie. Jednocześnie, minimalizacja przepływów między procesorami jest jednym z najważniejszych kryteriów dekompozycji.

Granica człowiek-komputer jest może najlepszym przykładem kosztu podejmowanych decyzji. Dla każdego przecięcia się jej z przepływem danych wejściowych lub wyjściowych jest konieczne dobranie właściwych środków technicznych, zaprojektowanie interfejsu użytkowego, opracowanie kształtu ewentualnych formularzy i raportów. (Należy zwrócić uwagę, że granica człowiek-komputer „powstaje" nie tylko w wyniku przydzielenia wybranych zadań procesorowi *Człowiek*, ale także na styku systemu ze światem zewnętrznym, reprezentowanym przez schemat kontekstu).

Problemy projektowania interfejsu człowiek-komputer, jakkolwiek integralne z omawianymi zagadnieniami, są na ogół traktowane odrębnie; nie będą też omawiane w tym artykule. Zainteresowanego czytelnika odsyłamy do bogatej literatury przedmiotu, m.in. [14,18].

LITERATURA

[1] Avison D. E. and Fitzgerald G.: Information Systems Development: Methodologies, Techniques and Tools. Blackwell Scientific Publications, 1988

[2] Coad P. and Yourdon E.: Object-Oriented Analysis/Design Yourdon Press, 1991

[3] Cohen B., Harwood W. T., Jackson M. I.: The Specification of Complex Systems. Addison-Wesley, 1986

[4] Curtis G.: Business Information Systems. Analysis, Design and Practice. Addison-Wesley Publ. Comp., 1989

[5] Cysewski G., Łyskawa H., Misiak D., Nowak P., Piechówka M., Szejko S.: Design prototyping: can it be a remedy? Materiały konf. EUROMICRO '95, Como, Italy. IEEE Computer Society Press, 1995

[6] Date C. J.: An Introduction to Database Systems, Addison-Wesley Publ Co, 1986

[7] DeMarco T.: Structured Analysis: Systems Specifications. Prentice-Hall, 1979.

[8] Dunn R.: Software Quality Assurance, Prentice-Hall, 1990

[9] ESA Software Engineering Standards, ESA PSS-0-0 Issue 2, 1991

[10] Farmer R.: Vending Machine: A Case Study in Teamwork. De Montfort Unversity, Leicester 1995

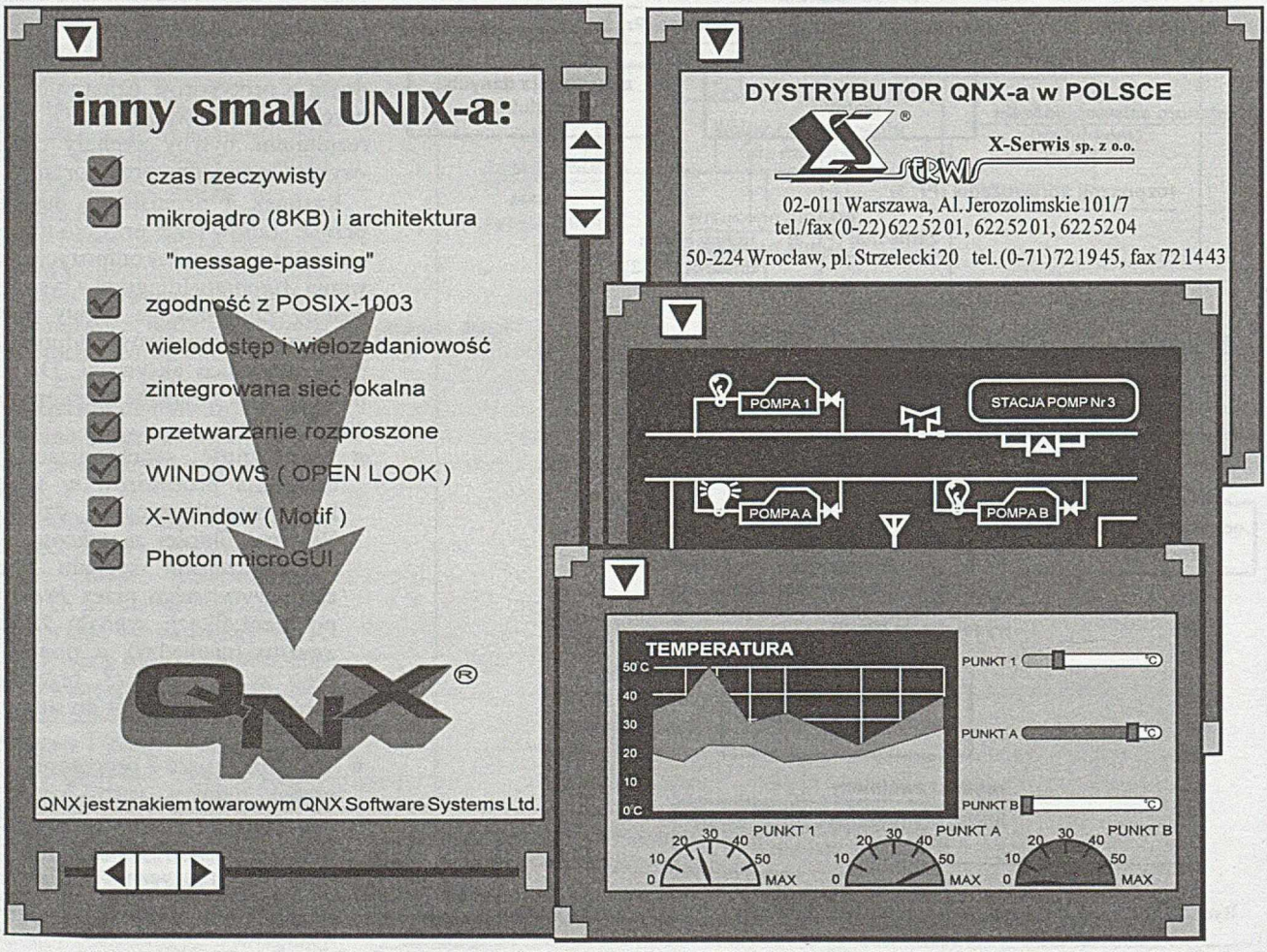
[11] Goldsmith S.: A Practical Guide to Real-Systems Development. Prentice-Hall Int., 1993

[12] Harel D., Pnueli A., Schmidt J. P. and Sherman R.: On the formal semantics of statecharts. Proc. Symposium on Logic in Computer Science, New York, June 1987

[13] Hatley D. J., Pirbhai I. A.: Strategies for Real-Time System Specyfication. Dorset House, 1987.

[14] Helander M. (ed.): Handbook of Human – Computer Interaction. North-Holland, Elsvier Sci. Publishers, 1991

[15] Kwiatkowski J., Misiak D., Szejko S.: Podejście strukturalne do analizy i projektowania systemów. Informatyka, 11–12, 1995



[16] Pollack S. L., Hicks H. T.(Jr), Harrison W.J.: Tablice decyzyjne. PWN, 1975
[17] Parkinson J.: Making CASE Work. NCC Blackwell Ltd., 1991
[18] Pressman R. S.: Software Engineering. A Practitioner's Approach. McGraw-Hill, Inc., 1992
[19] Reisig W.: Sieci Petriego. WNT, Warszawa, 1988
[20] Rumbaugh J., Blaha M., Premerlani W., Eddy S., Lorensen W.: Object-oriented Modeling and Design. Prentice-Hall 1991
[21] Sacha K.: Problemy jakości w projektowaniu oprogramowania systemów wbudowanych. Informatyka nr 9, 1995
[22] Sannella D.: A survey of formal software development methods. Intagrated Systems Factory Study rep., 1989
[23] Skidmore S., Farmer R. and Mills G.: SSADM Version 4. Models and Methods. NCC Blackwell Ltd., 1992
[24] Skidmore S. and Wroe B.: Introducing Systems Analysis/Design. NCC Blackwell 1980-1990
[25] Sommerville I.: Software Engineering. Addison-Wesley, 1992
[26] Ullman J.D.: Systemy baz danych. WNT, 1988
[27] Ward P. T. and Mellor S. J.: Structured Development for Real-Time Systems. Yourdon Press, 1985
[28] Wing J. M.: A specifier's introduction to formal methods. COMPUTER 1990, pp.8–24
[29] Yourdon E.: Modern Structured Analysis. Prentice-Hall, 1989.

Krajowe konferencje EDI

– próba oceny dorobku merytorycznego

Tegoroczna IV Krajowa Konferencja EDI odbędzie się w dniach 3-5 czerwca br. tradycyjnie w Arturówku k. Łodzi.

Trzy kolejne krajowe konferencje EDI (lata - 1993, 1994, 1995) zaowocowały m.in. opublikowaniem kilkudziesięciu referatów krajowych i zagranicznych, których charakterystyka ujęta została poniższej.

Tematyka	Liczba referatów	
	krajowe	zagraniczne
Wprowadzenie do problematyki EDI	7	-
Problemy zastosowań EDI w handlu i logistyce	8	-
Problemy zastosowań EDI w transporcie	5	-
Problemy zastosowań EDI w turystyce	2	-
Problemy zastosowań EDI w sektorze publicznym	6	3
Problemy zastosowań EDI w bankowości	5	-
Problemy zastosowań EDI w przemyśle	3	1
Problemy budowy systemów EDI	16	2
Problemy kształtowania środowiska dla potrzeb EDI	6	1
Problemy infrastruktury telekomunikacyjnej	7	1
Problemy prawne EDI	3	1

Przedstawione dane umożliwiają sformułowanie odpowiedzi na następujące pytania:
- o czym pisano w poprzednich latach?
- w jakim stopniu poszczególne zagadnienia, związane z elektroniczną wymianą danych, były podejmowane przez Autorów referatów?

Odpowiedzi na powyższe pytania są o tyle istotne, iż materiały drukowane z okazji kolejnych krajowych konferencji EDI są jednym z bardzo nielicznych źródeł aktualnej informacji o elektronicznej wymianie danych dostępnych w języku polskim, (niektóre z referatów były powtórnie publikowane w formie artykułów w miesięczniku *Informatyka*).

Analiza przedstawionych danych umożliwia również wyciągnięcie następujących wniosków ogólnych.
- Zauważalną cechą charakterystyczną przedstawionej wyżej listy tematów referatów jest jej obszerność; w zasadzie wszystkie zagadnienia, podejmowane na zagranicznych konferencjach EDI, są „obecne" na naszej krajowej imprezie.
- Pomimo, iż problematyka EDI jest nowością w warunkach polskich (zwłaszcza, jeśli chodzi o praktyczne wdrożenia), zwraca uwagę duża liczba referatów przygotowanych przez krajowych ekspertów oraz fakt, iż problematyka polskich referatów pokrywa właściwie cały obszar tematyczny EDI (nie ma „białych plam" zarezerwowanych wyłącznie dla referentów zagranicznych).

Warto również podkreślić fakt, iż w naszej konferencji uczestniczą zarówno referenci wywodzący się ze środowiska akademickiego, jak i tzw. praktycy. Wydaje się być to bardzo cenną wartością. Przywołując bowiem znaną analogię, można porównać teorię i praktykę do dwóch wioseł w łodzi. Koncentrując się wyłącznie na teorii albo też wyłącznie na praktyce nie jesteśmy w stanie uzyskać istotnego postępu (efekt kręcenia się w miejscu). Dopiero teoria zapładniana doświadczeniami praktycznymi oraz praktyka wykorzystująca dorobek teorii dają szansę skutecznego osiągnięcia efektu, o który nam chodzi.

Marian Niedźwiedziński

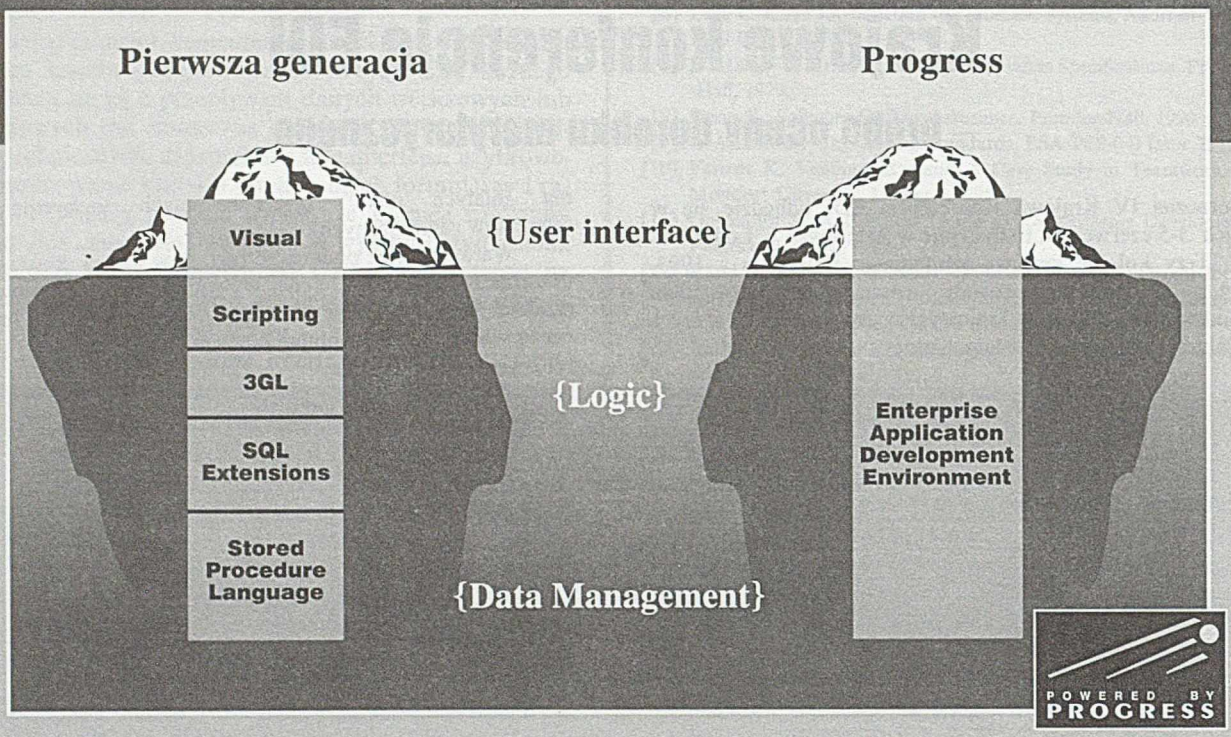
Informacji na temat konferencji udzielają:
Hanna i Marian Niedźwiedzińscy, tel. faks: (0-42) 70-72-86,
Eugeniusz Danikiewicz i Jan Durkiewicz,
tel. (0-22) 26-60-65 lub (0-22) 26-62-96
oraz faks: (0-22) 27-68-10.

Uwaga członkowie PTI!

W związku z licznymi pytaniami wpływającymi do redakcji od członków PTI, uprzejmie informujemy, że od stycznia br. w ramach porozumienia zawartego między naszą redakcją a Zarządem Głównym PTI, wszyscy członkowie tego stowarzyszenia mają uprawnienia do prenumerowania INFORMATYKI w cenie ulgowej, tj. 3,75 zł za egzemplarz. Wystarczy na blankiecie wpłaty wpisać „członek PTI". Nie są wymagane dodatkowe potwierdzenia tego faktu przez władze PTI.

Progress

Otwartość i elastyczność
Kompletne środowisko projektowe
Niezależność sprzętowa i systemowa



Systemy klient-serwer są jak góra lodowa: interfejs użytkownika, czyli to co jest dla wszystkich dobrze widoczne, w rzeczywistości stanowi jedynie małą część systemu informatycznego. Wraz ze wzrostem złożoności aplikacji rośnie znaczenie pozostałych części: logiki oraz danych. W dużych systemach, to właśnie "podwodna" część aplikacji decyduje o jej przydatności.

Tymczasem na rynku istnieje szereg narzędzi koncentrujących się głównie na obsłudze interfejsu użytkownika; aby oprogramować pozostałe elementy aplikacji trzeba korzystać z dodatkowych narzędzi i języków programowania, często pochodzących od różnych producentów. Łatwo sobie wyobrazić, jak trudno jest zapewnić dobrą współpracę tylu różnych komponentów.

Progress oferuje jedno zintegrowane środowisko do tworzenia wszystkich składowych aplikacji: *User Interface Builder* do tworzenia interfejsu, silny 4GL do programowania logiki oraz Słownik Danych i wysokowydajną bazę danych. Wszystkie elementy środowiska dostępne są w jednym pakiecie, umożliwiając szybkie tworzenie niezawodnych aplikacji o dowolnym stopniu złożoności.

Progress jest potężnym, graficznym, zintegrowanym środowiskiem do tworzenia i użytkowania przenośnych i skalowalnych aplikacji o najwyższych wymaganiach (mission-critical), pracujących w architekturze klient-serwer, host i mieszanej, oferującym kompletny zestaw narzędzi: bazę danych, rozbudowany słownik danych, generatory interfejsu i raportów, silny 4GL połączony z SQL oraz wyrafinowane narzędzia do wspomagania pracy grupowej, zarządzania konfiguracjami i instalacjami, zapytań i raportów. Współdziała z różnymi narzędziami CASE. Jest niezależny od interfejsu użytkownika, systemu operacyjnego, bazy danych i protokołu sieciowego. Działa na prawie 1500 różnych typach komputerów i kilkudziesięciu systemach operacyjnych, włączając UNIX (wszystkie odmiany RISC i Intel), Novell Netware, Windows, Windows NT, Windows 95, DOS, OS/2, OS/400 i inne.

Computer Systems for Business International S.A.

02-119 Warszawa, ul. Pruszkowska 17, tel: (0-22) 659-73-56, 659-04-16, fax (0-22) 659-04-85, 22-82-35
BHT 40-955 Katowice, ul. Bytkowska 1b, tel. (0-32) 153-72-79, fax (0-32) 154-22-65
BHT 53-332 Wrocław, ul. Powstańców Śląskich 5, tel. (0-71) 60-55-07, fax (0-71) 60-57-11
BHT 70-540 Szczecin, ul. Gen. Dąbrowskiego 38/40, tel./fax (0-91) 836-964

CSBI

O/10/96

informacje

Nowy C++

Borland na konferencji prasowej w Warszawie (27 lutego br.) przedstawił swoje najnowsze produkty oraz strategię dla Polski. Gerhard W. Rowen – nowy wiceprezes Borlanda na Europę Środkową i Wschodnią, obszar Morza Śródziemnego oraz Środkowy Wschód – oświadczył, że obecna strategia dla Polski to dostarczanie produktów i technologii Borlanda oraz usług poprzez wysoko wykwalifikowanych partnerów, którzy zapewnią klientom niezbędną wiedzę. Pracownicy oddziału w Niemczech mają za zadanie wspomagać partnerów, oferując im pomoc marketingową, techniczną i logistyczną. W Polsce partnerami Borlanda są ABC Data, Computer 2000, MSP oraz SoftPoint.

Na rynku europejskim pojawia się grupa najnowszych produktów Borlanda, m.in. C++ 5.0 dla Windows 95 i NT, który zawiera obecnie zintegrowane narzędzia do tworzenia programów w języku Java, w tym Java Development Kit (JDK) firmy Sun. Ponadto Borland przedstawił trzy nowe wersje oprogramowania Delphi 2.0 dla Windows 95 i NT.

C++ 5.0 umożliwia tworzenie aplikacji 32- i 16-bitowych oraz programów w języku Java. C++ 5.0 współpracuje zarówno z 16-, jak i z 32-bitowymi platformami (Windows 95, Windows NT, Windows 3.1 i DOS), użytkownicy mogą poruszać się łatwo pomiędzy różnymi systemami wykorzystując do tego celu TargetExpert, który pozwala na zmianę platformy za pomocą zwykłego kliknięcia myszą. Nowa wersja C++ umożliwia także równoczesne tworzenie 32- i 16-bitowych wersji tej samej aplikacji. C++ 5.0 jest w pełni kompatybilny z Windows 95 i jego standardami, takimi jak OLE, można m.in. tworzyć długie nazwy plików.

C++ 5.0 posiada nową wersję Object Windows Library (OWL), Microsoft Foundation Classes (MFC), ObjectScripting (nowy w pełni programowalny i elastyczny system umożliwiający modyfikowanie i konfigurowanie zintegrowanego środowiska twórczego IDE) oraz wsparcie dla 32- i 16-bitowych sterowników VBX. OWL 5.0 zawiera pojedyncze kody źródłowe oparte na 32- i 16-bitowych strukturach aplikacji windowsowych oraz wiele nowych API m.in. WinSock (TCP/IP) i MAPI (e-mail).

Cena rynkowa C++ 5.0 wynosi 349,95 dolarów, cena promocyjna, która obowiązuje do 31 maja to 299,95 dolarów. Użytkownicy poprzednich wersji C++ będą mogli dokonać upgrade-u za 249,95 dolarów (do 31 maja za 199,95).

Borland zamierza również sprzedawać C++ Developement Suite 5.0, w którym znajdą się C++ 5.0, CodeGuard 32/16, PVCS Version Manager, InstallShield Express oraz kompilator AppAccelerator for Java. Cena całego pakietu to 499,95 USD, do 31 maja można będzie go kupić za 299,95 USD. Użytkownicy poprzednich wersji C++ będą mogli kupić cały pakiet za 349 USD.

W tym kwartale pojawią się na rynku trzy nowe wersje Delphi 2.0, każda z nich zaprojektowana dla konkretnej grupy twórców oprogramowania: Delphi Client/Server Suite 2.0, Delphi Developer 2.0 (dla twórców oprogramowania opartego na sieciach lokalnych LAN) oraz Delphi Desktop 2.0.Delphi 2.0 zawiera kompilator rodzimych 32-bitowych kodów, narzędzia do programowania skalowalnych baz danych oraz bibliotekę komponentów drag-and-drop (przeciągnij-i-upuść), a wszystko to w obiektowo zorientowanym środowisku twórczym. Nowe wersje Delphi 2.0 są w pełni kompatybilne z nowymi właściwościami Windows 95 i Windows NT. Delphi 2.0 wspiera Microsoft OLE Controls (OCX), łączenie (*connectivity*) otwartych baz danych (ODBC) oraz dodatkowo technologię firmy Microsoft Remote Automation i nadchodzące Network OLE. Aby ułatwić migrację z 16- do 32-bitowych systemów, każda z wersji Delphi 2.0 zawiera również 16-bitową wersję Delphi 1.0 for Windows.

Delphi Desktop 2.0 kosztować będzie 495 USD, upgrade natomiast 299, a do 31 maja 169 USD. Ceny Delphi Developer 2.0 kształtują się odpowiednio na poziomie 815, 329 i 279 USD, natomiast cena Delphi Client/Server Suite 2.0 wynosi odpowiednio 2 629, 1 439 i 1 139 USD. **(aj)**

37 nowych modeli Compaqa

34 komputery z procesorem Pentium i trzy komputery z procesorami PentiumPro wprowadził w marcu br. na rynek Compaq. Oferowane po konkurencyjnych cenach komputery PC z Pentium, od modeli 75 MHz i 150 MHz, wprowadzono do serii ProLinea i Deskpro. Seria Deskpro XL zawiera również systemy z procesorami PentiumPro 150 MHz i 200 MHz, natomiast serię ProLinea wzbogacono o system PentiumPro 150 MHz. Deskpro XL 6200, wyposażony w najnowszy procesor Intela, PentiumPro 200 MHz, jest propozycją dla użytkowników 32-bitowych aplikacji oraz 32-bitowych systemów operacyjnych. Komputery Deskpro mają dyski twarde 840 MB, 1,2 GB i 2,0 GB oraz wbudowaną 32-bitową kartę sieciową PCI. Specjalnie przystosowane do środowiska sieciowego komputery Deskpro są wyposażone w zintegrowane sterowniki sieciowe, zgodne z DMI i obsługują Inteligentne Zarządzanie (*Intelligent Manageability*). Do systemów Deskpro i ProLinea są dodawane napędy dyskowe LS-120, które zostały opracowane przez Compaqa, 3M i Matsushita-Kotobuki. Jest to pierwszy napęd dyskietek o pojemności 120 MB, może on też odczytywać i zapisywać standardowe dyskietki 1,44 MB i 720 KB. **(k)**

Stacje robocze Zenith Data Systems

Firma Zenith Data Systems (ZDS) wprowadza na rynek stacje robocze Z-STATION GT wyposażone w procesory Pentium 150 MHz i 166 MHz. Jest to oferta adresowana do użytkowników, wykorzystujących komputer do takich prac, jak CAD/CAM, sporządzanie prognoz finansowych czy profesjonalne prace wydawnicze. Komputery Z-STATION GT wyposażono w 64-bitowe karty graficzne PCI, umożliwiające odtwarzanie plików wideo w standardzie MPEG, dyski o pojemności 1,6 GB. Pamięć podręczna pracuje w trybie „pipeline burst". Wszystkie modele mają firmowo zainstalowany Windows 95, przeglądarkę Netscape, Norton Nawigator oraz oprogramowanie multimedialne ZDS EZ-Companion, pomagające użytkownikowi w zapoznaniu się z możliwościami komputera. (k)

IV Ogólnopolska Konferencja TeX-owa

W dniach 2–4 maja 1996 r. w Ośrodku wypoczynkowym Uniwersytetu Mikołaja Kopernika w Bachotku odbędzie się IV Konferencja TeX-owa BachoTeX '96 pn. Świat wokół TeX-a. Poświęcona będzie różnym aspektom publikacji elektronicznych i ogólnym zagadnieniom typografii oraz problemom związanym z TeX-em, METAFONT-em i PostScriptem. Bliższych informacji na temat konferencji udziela Tomasz Przechlewski, Uniwersytet Gdański, ul. Armii Krajowej 119/121, 81-824 Sopot (**e-mail: gust@ifpan.edu.pl**). (k)

ComNet Warsaw '96

W dniach 19–21 czerwca 1996 r. odbędą się już po raz drugi w Warszawie targi ComNet, których tematyka jest związana z sieciami komputerowymi i telekomunikacją. Swój udział w tegorocznej edycji targów zapowiedziały takie firmy sieciowe i komunikacyjne, jak Netscape, Nokia, Siemens, Tele Denmark International i wiele innych. Oprócz możliwości nawiązania kontaktów handlowych, uczestnicy ComNetu będą mogli uczestniczyć w licznych seminariach, na których będą omawiane zagadnienia związane z ISDN, ATM, GSM, Internetem, komunikacją cyfrową, bezpieczeństwem sieci i innymi ważnymi problemami świata komunikacji. (k)

Drukarka laserowa w cenie atramentowej

Firma OKI przedstawiła na targach CeBIT laserową drukarkę Okipage 4w, która ceną konkuruje z drukarkami atramentowymi (w Polsce ma kosztować poniżej tysiąca zł). Obniżenie ceny jest możliwe dzięki zastosowaniu opracowanej przez OKI techniki LED i systemu GDI. W technice LED głowicę laserową zastępuje listwa ciasno upakowanych diod. Okipage 4w jest przeznaczona dla rynku SOHO (*Small Office/Home Office*). Ma rozdzielczość 600 dpi, drukuje cztery strony na minutę, do pracy jest gotowa po 23 sekundach. Adresowana jest wyłącznie do posiadczy Windows 95 i Windows 3.1; użytkownicy DOS mogą z niej korzystać przez okienko DOS-owe w Windowsach. Drukarka ma też inne walory dla indywidualnych użytkowników i małych biur: wielkość kartki formatu A4 i waży niecałe 4 kg. Okipage 4w posiada podajnik na 100 stron papieru, ręcznie można podawać koperty i nietypowy papier. Producent ocenia żywotność drukarki na pięć lat lub 30 tys. stron druku. (k)

16-bitowy przetwornik analogowo-cyfrowy dla VME

PEP Modular Computers Sp. z .o.o. wprowadził do swojej oferty zaawansowany przetwornik analogowo-cyfrowy w standardzie VME pod nazwą VADI-3. Zapewnia on konwersję o wysokiej precyzji przy niskich kosztach. Karta jest wyposażona w 16 analogowych kanałów różnicowych podzielonych na dwie grupy. Każda z nich może zostać wyposażona w specjalną nakładkę dopasowującą rodzaj interfejsu analogowego do potrzeb użytkownika. Wbudowany w przetwornik układ A/C z konwersją Delta-Sigma zapewnia rozdzielczość 16 bitów przy czasie konwersji 125 ms. Karta spełnia standardowe wymagania i zapewnia pełną izolację galwaniczną, pracę w rozszerzonym zakresie temperatury od −40° C do 85° C, wysoką odporność na drgania, mały pobór mocy oraz pracę bez wymuszonego chłodzenia. (t)

*

Dwa nowe multipleksery inwersyjne IMX-4E1 i IMX-4T1, które wykorzystują szeroko dostępne łącza WAN T1 i E1, wprowadziła na rynek firma RAD. Użytkownicy przesyłający interakcyjne obrazy, korzystający z programów multimedialnych i innych wymagających dużych prędkości przesyłania, którzy potrzebują tylko 2 do 8 Mbps, nie będą dzięki nim płacić za łącza 45 Mbps E3 czy 34 Mbps T3. Jest to również atrakcyjne rozwiązanie dla operatorów sieci, ponieważ umożliwia im dostarczanie najbardziej poszukiwanej usługi bez konieczności rozbudowy sieci szkieletowej.

*

Firma Bentley Systems – twórca środowiska projektowania CAD/CAM/CAE/GIS Microstation, w styczniu br. uruchomiła swoje przedstawicielstwo w Polsce. Tymczasowa siedziba firmy mieści się we Wrocławiu (tel.-faks: 0-71 65 20 55, e-mail: tomasz.puchalski@bentley.nl. Głównym celem BS Polska jest wspieranie autoryzowanych biznes partnerów w ich działalności dystrybucyjnej.

*

3Com proponuje nowy zestaw produktów ISDN, które pozwalają na realizację połączeń zdalnych dla kompletnych systemów końcowych. Nowe produkty to Aperature II, urządzenie umożliwiające zdalne połączenia, 3Com AccessBuilder 4000 ISDN serwer zdalnego dostępu i 3Com Apreggio most/router. Pojedynczy 3Com Aperature II, który zawiera na jednej platformie terminal, most i modem analogowy w wersji ISDN, może obsłużyć do 10 interfejsów PRI (*Primary Rate Interface*). Terminal oraz modem są programowo konfigurowalne, co zapewnia wysoką wydajność cyfrowych kart procesowych.

Kadra naukowa pilnie poszukiwana

dokończenie z pierwszej strony okładki

Powody są takie same, jak w przypadku wyższych uczelni; śmiesznie małe, w porównaniu z oferowanymi przez firmy, płace.

Informatyka jest nie tylko nauką młodą, ale i nieustannie rozwijającą się. Programy nauczania muszą być ciągle modyfikowane, rynek wymusza bardzo szybkie zmiany. Wszystko to sprawia, że uczestnicy I Kongresu Informatyki Polskiej ostrzegali: *Obecne struktury studiów oraz programy nauczania w państwowych uczelniach nie zawsze gwarantują przygotowanie odpowiednio wykształconych kadr na potrzeby użytkowników i firm informatycznych... Uczelnie bez planów reformy wewnętrznej oraz reformy ich efektywnego finansowania nie mają szans uniknąć w najbliższych czasach opisywanego zagrożenia...W XXI wiek więc wejdziemy z ogromnym brakiem umiejętności zrozumienia tego, co nam będzie oferował świat.*

Nasze pismo chciałoby pokazywać, jak mimo tych trudności polskie uczelnie starają się dotrzymać kroku rozwojowi informatyki i podtrzymać dobre imię polskiej szkoły informatycznej. Uznanie, jakim cieszyła się ona w latach siedemdziesiątych i osiemdziesiątych sprawiło, że chociaż nie było w Polsce rynku informatycznego, mieliśmy dostęp do wielu informacji w zagranicznych centrach badawczych i dydaktycznych, zaś absolwenci polskich uczelni zatrudniani są na najbardziej prestiżowych uczelniach świata i we wszystkich znaczących firmach informatycznych. Dyskusję na temat edukacji informatycznej, zmian, jakie wywołuje w jej nauczaniu nasz rozwijający się rynek informatyczny, zaczynamy od rozmowy z profesorem Janem Madeyem. Mamy nadzieję, że głos na ten temat zabiorą zarówno przedstawiciele wyższych uczelni, jak i szefowie firm, które zatrudniają absolwentów wydziałów informatycznych. Potrzebna jest, jak sądzę, taka wymiana doświadczeń i poglądów.

Krystyna Karwicka

Intel zaprezentował nową serię kart pamięci błyskowej o pojemnościach 2,4 i 8 MB, które są najtańszymi wymiennymi nośnikami pamięci w standardzie PCMCIA. Karty te nadają się do zastosowań sieciowych i telekomunikacyjnych oraz do emulacji szybkich pamięci dyskowych w komputerach przenośnych. Karty są oparte na modelu liniowym, w którym układy scalone pamięci błyskowej są zamontowane na karcie praktycznie bez innych dodatkowych modułów logicznych.

*

W plebiscycie magazynu Datamation (*Technology & Business Solutions for IS Managers*) na Produkty Roku 1995 w kategorii Data Warehouse zwyciężyło rozwiązanie oferowane w systemie SAS. W uzasadnieniu redakcja Datamation, które uchodzi za pismo menażerów informatycznych, podaje, że w 1995 r. SAS dodał do swojego systemu solidne narzędzia do ekstradycji danych, ich dystrybucji, walidacji i reformatowania. Systemy Data Warehouse wdrożone w Systemie SAS służą jako podstawa systemów informacyjnych typu MIS (*Management Information System*), EIS (*Executive Information System*) i DSS (*Decision Support System*).

*

Western Digital wzbogaca swoją rodzinę napędów Caviar EIDE o dwa nowe napędy wysokiej pojemności: Caviar AC32500 (3 dyski, 2,5GB) i Caviar AC32100 (3 dyski, 2,1GB). Napędy Caviar działają zarówno w trybie przesyłania danych Mode 4PIO – programowalnego wejścia wyjścia, jak i w trybie Mode 2 DMA – bezpośredniego dostępu do pamięci o szybkościach przesyłania danych do 16,6 megabajtów na sekundę, przy prędkości 5200 obrotów na minutę. Prędkość ta, zapewniająca zmniejszoną latencję mechaniczną oraz osiąganie stałych szybkości przesyłania danych, znacznie poprawia wydajność komputerów PC.

*

Firma Corel poinformowała, iż zawarła z Novellem porozumienie dotyczące nabycia rodziny programów WordPerfect i innych związanych z nią technologii. Corel będzie w stanie wprowadzić na rynek istniejące produkty firmy dla bazy użytkowników szacowanej na ponad 20 mln osób, uzyska także dostęp do nowych możliwości w zainstalowanej bazie 55 mln użytkowników Novella. Novell i Corel współpracują nad dalszą integracją zestawu PerfectOffice z oprogramowaniem typu klient systemu GroupWise firmy Novell i z oprogramowaniem typu serwer systemu NetWare.

*

Autodesk przed kilkoma miesiącami uruchomił w Internecie swoją stronę World Wide Web. Pod adresem **http:\\www.autodesk.com** można znaleźć przeglądarkę dostosowaną do komputerów stacjonarnych i przenośnych, dostarczającą informacji użytkownikom oraz tym, którzy dopiero planują zakup produktów firmy. Codziennie rejestruje się ok. 15 tys. połączeń, z czego ok. 25% to połączenia z dużymi korporacjami, 20% z instytucjami edukacyjnymi, a 30% z przedsiębiorstwami prowadzącymi usługi on-line. Autodesk ma także węzeł FTP (**ftp.autodesk.com**), z którego można skopiować wiele programów, danych, uzupełnień i pomocy. Tygodniowo są kopiowane programy o objętości ponad pięć terabajtów.

*

Silicon Graphics przejął większość udziałów Cray Research. W ciągu ostatnich 18 miesięcy Cray został zrestrukturyzowany i dzięki wprowadzeniu nowych produktów, takich jak równoległy system wektorowy CRAY T90 oraz skalowalne systemy CRAY T3D i CRAY T3E, ponownie zaczął przynosić zyski. Przejęcie Craya przyczyni się do rozszerzenia oferty Silicon Graphics o superkomputery, których moc jest liczona w teraflopsach. Prezes Craya uważa, że zaowocuje to nie tylko najpotężniejszymi na świecie komputerami, ale również najbardziej atrakcyjnym wskaźnikiem cena – wydajność.

*

Microsoft wprowadził nowe zasady udzielania pomocy technicznej dla produktów z rodziny BackOffice. Pomoc techniczna realizowana dotychczas przez biura pomocy technicznej Microsoftu została przeniesiona do firmy Digital Equipment Polska, która ma status Autoryzowanego Ośrodka Wspomagania Technicznego Microsoft. Digital udziela płatnej pomocy technicznej i zawiera kontrakty serwisowe obejmujące m.in. serwis oprogramowania przez telefon (tel. 0-22 640-00-11), wizyty u klientów, instalowanie, konfigurowanie i administrację systemami, konsultacje.

*

Firma RAD Data Communications wprowadza nowy modem HTU-E1 do sieci HDSL. Powiększa on do 4,5 km zasięg stosowanych urządzeń E1, przez czterożyłowy kabel miedziany 24 AWG (średnicy 0,5 mm). Dzięki jego zastosowaniu nie jest konieczna instalacja wtórników czy używanie kabli światłowodowych na końcowych odcinkach sieci, tam gdzie istnieje już okablowanie miedziane. Modem HTU-E1 umożliwia również poprawną transmisję E1 przez krawędzie mostków, sieci o różnej przepustowości i linie o złej jakości.

*

Ethernetowy przełącznik grupowy EWS firmy CrossComm zdobył nagrodę honorową w rankingu wydawców Network Computing, przeprowadzoną wśród najnowszych produktów o cenie poniżej 250 USD za port. Wyróżniony przełącznik jest członkiem rodziny modułów przełączających, opartych na układach scalonych w technologii ASIC drugiej generacji. Wyposażony w sondę RMON i agenta SNMP, cechuje się wysoko zaawansowaną skalowalnością.

Sprostowanie

W styczniowym numerze Informatyki do informacji o firmie Informix Corp. zakradł się niefortunny błąd: przychód netto na akcję został podany jako cena akcji Informixa. Prawidłowa informacja powinna brzmieć: przychód netto na akcję wzrósł z 0,49 USD w 1994 r. do 0,76 USD za rok 1995 czyli o 55%. Przepraszamy za pomyłkę.

Zaprosili nas...

Centrum Kongresowe EXBUD w Kielcach na dwudniową konferencję nt. wdrażania technologii informatycznych w przemyśle – ze szczególnym uwzględnieniem metody MRPII (Material Requirement Planning).

2SI, czyli Sieciowe Systemy Informacyjne, na seminarium z okazji zmiany nazwy firmy.

Centrum Prasowe PAI w imieniu Polskiego Komitetu Normalizacyjnego, Polskiego Centrum Badań i Certyfikacji oraz polskiego Forum ISO 9000 na konferencję połączoną z wręczeniem certyfikatów Systemu Jakości.

Polska OnLine Sp. z o.o. na inaugurację Klubu Użytkowników Polska OnLine (w skrócie KUPOL) otwierającą cotygodniowe spotkania środowiska komputerowo-informatycznego Warszawy.

Compaq Computer Sp. z o.o. na konferencję informującą o wynikach finansowych firmy oraz obsłudze Internetu na serwerach Compaqa.

Acer Computer na konferencję prasową z okazji otwarcia w Polsce przedstawicielstwa firmy.

Fundacja Kultury Informatycznej wraz z Oficyną Informatyczną na otwarte spotkanie czwartkowe, poświęcone w kwietniu zmianom w szkoleniach informatycznych w świetle rozwoju zastosowań informatyki.

Novell Polska Sp. z o.o. na seminarium nt. NetWare 4.1 Inside Out.

IDG Poland S.A. na spotkanie zapowiadające II Międzynarodową Konferencję i Wystawę Technik Telekomunikacyjnych i Sieci Komputerowych ComNet '96.

Do Hanoweru na CeBIT '96:

3Com na konferencję prasową i prezentację nowej serii produktów.

APC na party.

INWAR S.A. do odwiedzenia stoiska.

Zakład Elektroniki Górniczej ZEG S.A. w Tychach oraz

Instytut Informatyki Politechniki Śląskiej w Gliwicach do zwiedzenia wspólnego stoiska.

Austrade – Australijska Komisja Handlowa działająca przy Ambasadzie Australii w Polsce do obejrzenia stoisk swoich 100 firm.

Polska Izba Informatyki i Telekomunikacji oraz Państwowa Agencja Inwestycji na seminarium pn. „Inwestycje na polskim rynku teleinformatycznym".

Do Kielc na INTERTELECOM '96

Alcatel Polska do wzięcia udziału w seminarium.

Do Poznania na INFOSYSTEM '96

INITEL-Serwis do odwiedzenia stoiska.

Oracle Polska na dwa seminaria.

Instytut Informatyki Politechniki Poznańskiej i Poznańskie Centrum Superkomputerowo-Sieciowe na III Konferencję pn. „Miejskie Sieci Komputerowe w Nauce, Gospodarce i Administracji POLMAN'96". (t)

informatyka

Miesięcznik
ISSN 0542-9951
INDEKS 36124

Uwaga, dzięki gościnności Polska Online ukazuje się już elektroniczna wersja naszego miesięcznika. Oto nasz adres w Internecie:
http://www.pol.pl/informatyka/

nr 4 1996
Kwiecień
Rok wydania XXXII

REDAGUJE ZESPÓŁ:
mgr Krystyna **KARWICKA-RYCHLEWICZ** (redaktor naczelny)
mgr Jerzy **SZYLLER** (z-ca redaktora naczelnego)
mgr Teresa **JABŁOŃSKA** (sekretarz redakcji))
Przemysław **BASZKIEWICZ** (redaktor)
mgr Zdzisław **ŻURAKOWSKI** (redaktor działu)
Alina **KLEPACZ** (sekretariat)

KOLEGIUM REDAKCYJNE:
prof. dr hab. Leonard BOLC,
mgr inż. Piotr FUGLEWICZ,
prof. dr hab. Jan GOLIŃSKI,
dr inż. Zenon KULPA,
prof. dr inż. Jan MULAWKA,
prof. dr hab. Wojciech OLEJNICZAK,
mgr inż. Jan RYŻKO,
dr Witold STANISZKIS,
dr inż. Jacek STOCHLAK,
prof. dr hab. Maciej STOLARSKI,
prof. dr hab. Zdzisław SZYJEWSKI,
prof. dr hab. inż. Ryszard TADEUSIEWICZ,
prof. dr hab. Jan WĘGLARZ

PRZEWODNICZĄCY RADY PROGRAMOWEJ
prof. dr hab. Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek Technicznych SIGMA NOT Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
00-950 Warszawa,
ul. Ratuszowa 11, p. 628 i 644
skrytka pocztowa 1004
tel., fax: 619-11-61
tel.: 619-22-41 w. 159
e-mail:
informat@pol.pl

Materiałów nie zamówionych redakcja nie zwraca

Autorzy artykułów proszeni są o przysyłanie tekstów na dyskietkach $3^1/_2$" – w czystych kodach ASCII (Latin II, Mazovia) lub edytorach: Word 2, Word 6 – oraz odbitki na papierze. Teksty nie mogą przekraczać 20 tys. znaków.

Redakcja nie ingeruje w treść i formę ogłoszeń i innych materiałów reklamowych, w związku z tym nie ponosi za nie odpowiedzialności.

Ogłoszenia przyjmują:
- **Redakcja**
- **Agencja reklamowa KUBA**
  ul. Kopernika 28 m 23
  00-336 Warszawa
  tel.: 27-98-66, faks: 27-24-91

lub
- **Dział Reklamy i Marketingu**

00-950 Warszawa, ul. Mazowiecka 12
tel.: 27-43-66, faks: 26-80-16

W numerze:

Strona

Od teorii do praktyki – Z profesorem dr. hab. Janem Madeyem rozmawia *Krystyna Karwicka* 1

publikacje

Zastosowanie komputera do automatycznego układania rozkładu zajęć dla szkoły wyższej – *Beata Jendrzejczyk, Marek Skomorowski* 2

Wprowadzenie do systemów uczących się ze wzmocnieniem – *Paweł Cichosz* 4

Zagadnienia pozyskiwania wiedzy dla sterowników rozmytych (2) – *Jarosław Stańczak* 11

Ewolucja metod zarządzania pamięcią w dużych systemach komputerowych – *Josch Krischer* 15

Systemy czasu rzeczywistego

Metody projektowania oprogramowania. Projektowanie strukturalne systemów czasu rzeczywistego (1) – *Jan Kwiatkowski, Stanisław Szejko* 24

informacje

Nowy C++ – (*aj*) 33

W najbliższych numerach:

- **Marek Prokop omawia możliwości wykorzystania cech obiektowych języka Progress 4GL w oprogramowaniu relacyjnych baz danych.**
- **Marcin Gorawski i Andrzej Konopacki podają kompendium wiedzy na temat Magic-a, generatora aplikacji baz danych.**
- **Andrzej Wolski charakteryzuje nowy język obiektowy Java.**
- **Jacek Sedel i Michał Kolano prezentują język SQLWindows – nowoczesne narzędzie do tworzenia aplikacji korzystających z baz danych.**
- **Maciej Piasecki opisuje koncepcję modelu dla obiektowej bazy danych.**

Warunki prenumeraty na 1996 r.

Zamówienia na prenumeratę czasopism wydawanych przez Wydawnictwo SIGMA-NOT można składać w dowolnym terminie. Mogą one obejmować dowolny okres czasu, tzn. dotyczyć dowolnej liczby kolejnych zeszytów każdego czasopisma.

Zamawiający może otrzymywać zaprenumerowany przez siebie tytuł począwszy od następnego miesiąca po dokonaniu wpłaty. Zamówienia na zeszyty sprzed daty otrzymania wpłaty będą realizowane w miarę możliwości – z posiadanych zapasów magazynowych.

Warunkiem przyjęcia i realizacji zamówienia jest otrzymanie z banku potwierdzenia dokonania wpłaty przez prenumeratora. Dokument wpłaty jest równoznaczny ze złożeniem zamówienia.

Wpłat na prenumeratę można dokonywać na ogólnie dostępnych blankietach w Urzędach Pocztowych (przekazy pieniężne) lub Bankach (polecenie przelewu), przekazując środki pod adres:
Wydawnictwo SIGMA-NOT Spółka z o.o.
Zakład Kolportażu
00-716 Warszawa, skr. poczt. 1004
konto:
PBK S.A. III 0/Warszawa nr 370015-1573-139-11

Wpłaty na prenumeratę przyjmują także wszystkie urzędy pocztowe nadawczo-odbiorcze oraz doręczyciele na terenie całego kraju

Na blankiecie wpłaty należy czytelnie podać nazwę zamawianego czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty oraz własny adres.
Na życzenie prenumeratora, zgłoszone np. telefonicznie, Zakład Kolportażu ul. Bartycka 20, 00-950 Warszawa, (telefony: 40-30-86, 40-35-89 oraz 40-00-21 wew. 249, 293, 299) wysyła specjalne blankiety zamówień wraz z aktualną listą tytułów i cennikiem czasopism.

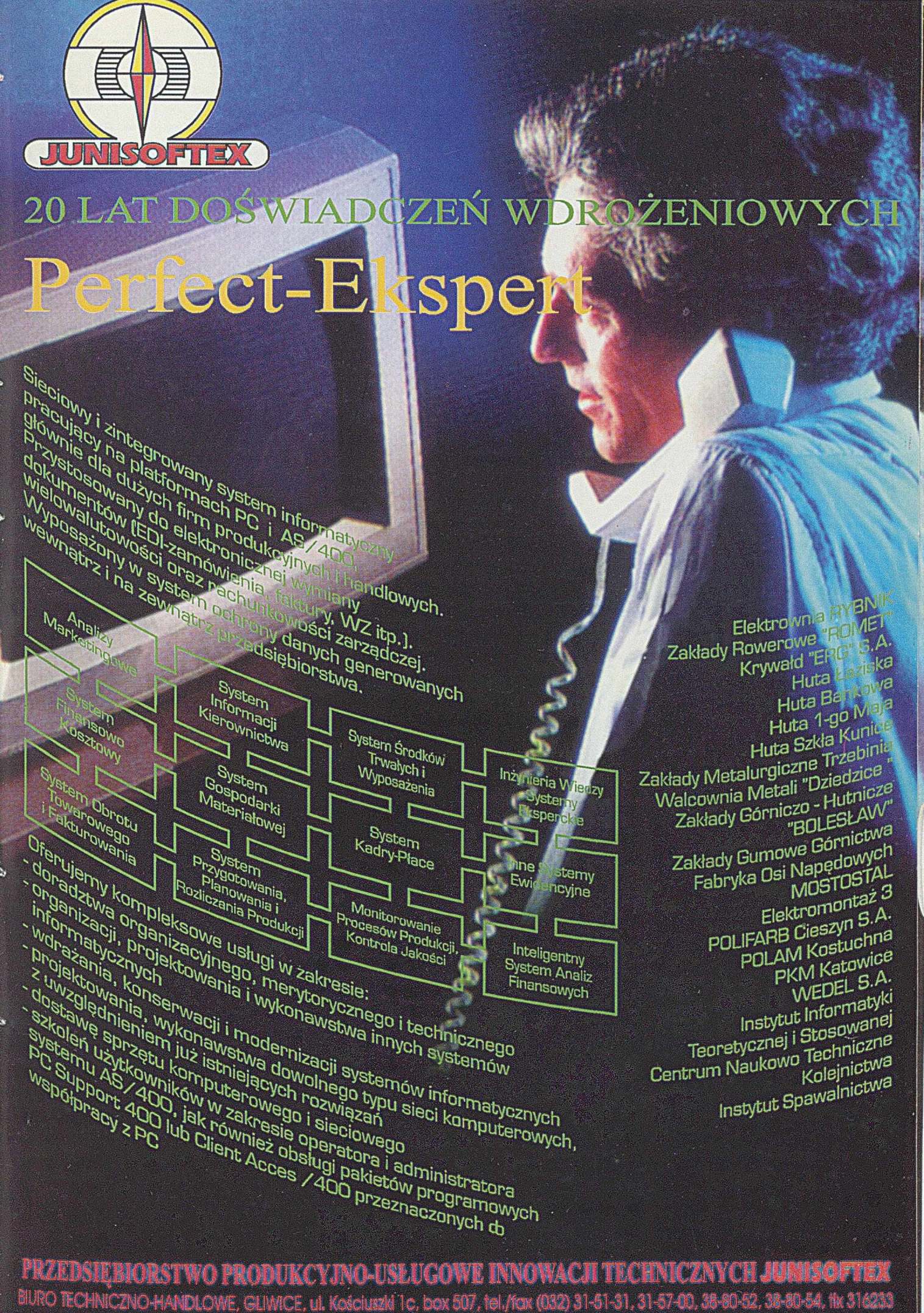
Odbiorcy zagraniczni mogą otrzymywać czasopisma poprzez prenumeratę dewizową (wpłata dokonywana poza granicami Polski w dewizach, wg cennika dewizowego z cenami podanymi w dolarach amerykańskich) lub poprzez zamówioną w kraju prenumeratę ze zleceniem wysyłki za granicę (zamawiający podaje dokładny adres odbiorcy za granicą, dokonując równocześnie wpłaty w wysokości dwukrotnie wyższej niż cena normalnej prenumeraty krajowej).

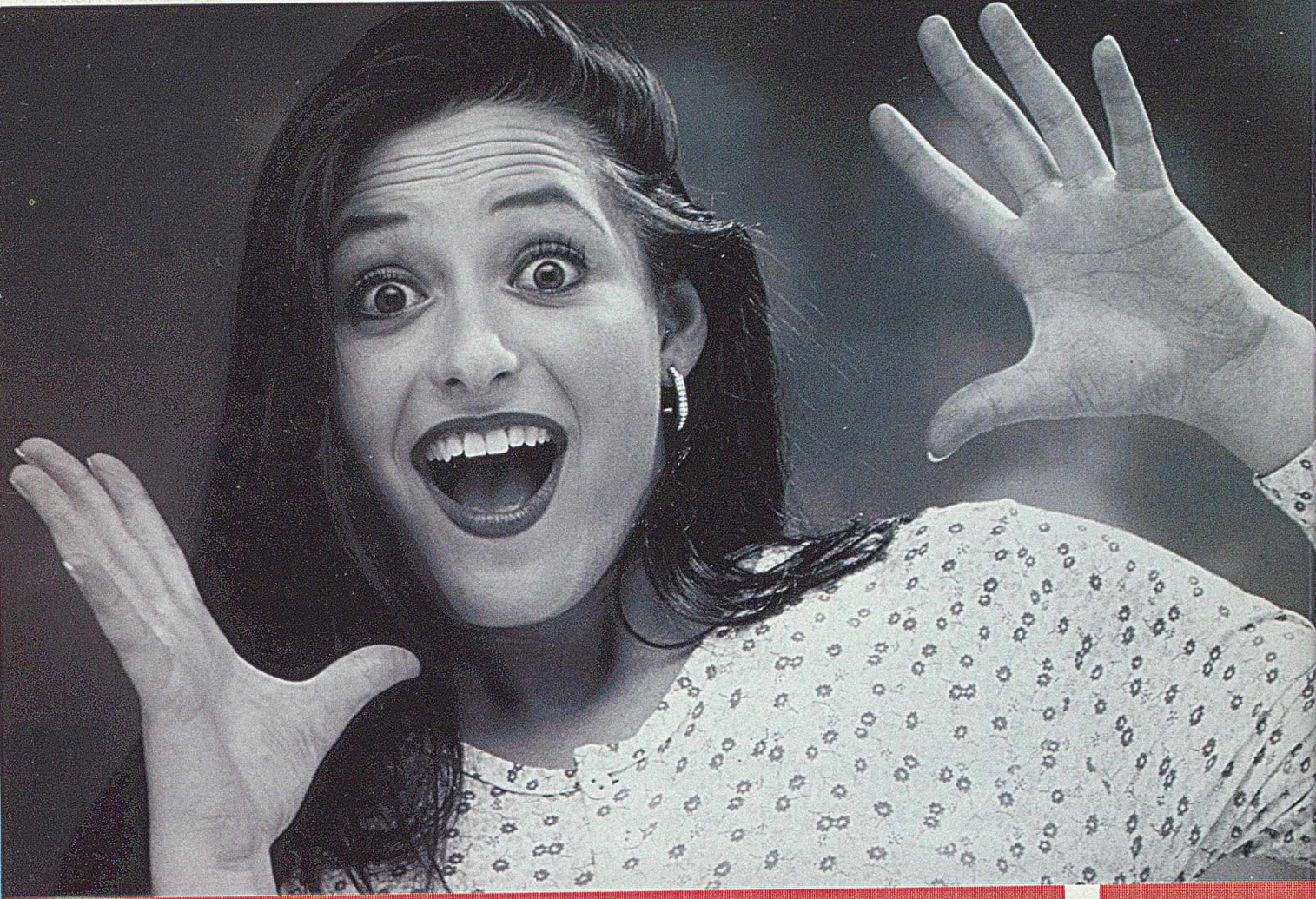
Egzemplarze archiwalne (sprzedaż przelewowa lub za zaliczeniem pocztowym) można zamawiać pisemnie, kierując zamówienia pod adresem: Wydawnictwo SIGMA NOT, Spółka z o.o. Zakład Kolportażu, 00-716 Warszawa, ul. Bartycka 20, paw. B, tel. 40-37-31, natomiast za gotówkę można je nabyć w Klubie Prasy Technicznej w Warszawie ul. Mazowieckiej 12, tel. 26-80-17.

Istnieje możliwość zaprenumerowania 1 egz. czasopisma po cenie ulgowej przez indywidualnych członków stowarzyszeń naukowo-technicznych zrzeszonych w FSNT oraz przez uczniów zawodowych i studentów szkół wyższych. Blankiet wpłaty na prenumeratę ulgową musi być opatrzony na wszystkich odcinkach pieczęcią koła SNT lub szkoły.

W przypadku zmiany cen w okresie objętym prenumeratą Wydawnictwo zastrzega sobie prawo do wystąpienia o dopłatę różnicy cen oraz prawo do realizowania prenumeraty tylko w pełni opłaconej.
Cena jednego egzemplarza: normalna 5,00 zł (50 000 zł), ulgowa 3,75 zł (37 500 zł)
Wartość prenumeraty w zł:
Normalna: kwartalna 15,00 zł (150 000 zł), półroczna 30,00 zł (300 000 zł), roczna 60,00 zł (600 000 zł)
Ulgowa: kwartalna 11,25 zł (112 500 zł), półroczna 22,50 zł (225 000 zł), roczna 45,00 zł (450 000 zł)

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 103/96


JUNISOFTEX
20 LAT DOŚWIADCZEŃ WDROŻENIOWYCH
Perfect-Ekspert
Sieciowy i zintegrowany system informatyczny pracujący na platformach PC i AS/400, głównie dla dużych firm produkcyjnych i handlowych. Przystosowany do elektronicznej wymiany dokumentów (EDI-zamówienia, faktury, WZ itp.). wielowalutowości oraz rachunkowości zarządczej. Wyposażony w system ochrony danych generowanych wewnątrz i na zewnątrz przedsiębiorstwa.
Analizy Marketingowe
System Finansowo Kosztowy
System Informacji Kierownictwa
System Gospodarki Materiałowej
System Obrotu Towarowego i Fakturowania
System Przygotowania, Planowania i Rozliczania Produkcji
System Środków Trwałych i Wyposażenia
System Kadry-Płace
Monitorowanie Procesów Produkcji, Kontrola Jakości
Inżynieria Wiedzy Systemy Ekspercke
Inne Systemy Ewidencyjne
Inteligentny System Analiz Finansowych
Oferujemy kompleksowe usługi w zakresie:
- doradztwa organizacyjnego, merytorycznego i technicznego
- organizacji, projektowania i wykonawstwa innych systemów informatycznych
- wdrażania, konserwacji i modernizacji systemów informatycznych
- projektowania, wykonawstwa dowolnego typu sieci komputerowych, z uwzględnieniem już istniejących rozwiązań
- dostawę sprzętu komputerowego i sieciowego
- szkoleń użytkowników w zakresie operatora i administratora systemu AS/400, jak również obsługi pakietów programowych PC Support 400 lub Client Acces /400 przeznaczonych do współpracy z PC
Elektrownia RYBNIK
Zakłady Rowerowe "ROMET"
Krywałd "ERG" S.A.
Huta Łaziska
Huta Bankowa
Huta 1-go Maja
Huta Szkła Kunice
Zakłady Metalurgiczne Trzebinia
Walcownia Metali "Dziedzice"
Zakłady Górniczo - Hutnicze "BOLESŁAW"
Zakłady Gumowe Górnictwa
Fabryka Osi Napędowych
MOSTOSTAL
Elektromontaż 3
POLIFARB Cieszyn S.A.
POLAM Kostuchna
PKM Katowice
WEDEL S.A.
Instytut Informatyki Teoretycznej i Stosowanej
Centrum Naukowo Techniczne Kolejnictwa
Instytut Spawalnictwa
PRZEDSIĘBIORSTWO PRODUKCYJNO-USŁUGOWE INNOWACJI TECHNICZNYCH JUNISOFTEX
BIURO TECHNICZNO-HANDLOWE, GLIWICE, ul. Kościuszki 1c, box 507, tel./fax (032) 31-51-31, 31-57-00, 38-80-52, 38-80-54, tlx 316233



Aż tak mała?!

OKIPAGE 4w

drukarka stronicowa

Tak - i możesz ją mieć!

Tak mała, bo mniejsza od zwykłej kartki papieru formatu A4, a drukuje z dokładnością lasera i rozdzielczością 600 dpi. Dzięki specjalnym systemom oszczędzania energii i tonera oraz trwałości elementów (5 lat gwarancji na głowicę) jest tania w eksploatacji. **Tak mało** kosztuje - bo tylko 999 zł. Przeznaczona do pracy w środowisku Windows 3.1, Windows 95. Tak tania, tak mała i tak dobra **za 999 zł - i możesz ją mieć.**

OKI

People to People Technology